Machine learning classification of plant genotypes grown under different light conditions through the integration of multi-scale time-series data

by

Nazmus Sakeef

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

In order to mitigate the effects of a changing climate, agriculture requires more effective evaluation, selection, and production of crop cultivars in order to accelerate genotype-to-phenotype connections and the selection of beneficial traits. Critically, plant growth and development are highly dependent on sunlight, with light energy providing plants with the energy required to photosynthesize as well as a means to directly intersect with the environment in order to develop. In plant analyses, machine learning and deep learning techniques have proved ability to learn plant growth patterns, including detection of disease, plant stress, and growth using a variety of image data. To date, however, studies have not assessed machine learning and deep learning algorithms for their ability to differentiate a large cohort of genotypes grown under several growth conditions using time-series data automatically acquired across multiple scales (daily and developmentally). Here, we extensively evaluate a wide range of machine learning and deep learning algorithms for their ability to differentiate 17 well-characterized photoreceptor deficient genotypes differing in their light detection capabilities grown under several different light conditions. Using algorithm performance measurements of precision, recall, F1-Score, and accuracy, we find that Suport Vector Machine (SVM) maintains the greatest classification accuracy, while a combined ConvLSTM2D deep learning model produces the best genotype classification results across the different growth conditions. Critically, our successful integration of time-series growth data across multiple scales, genotypes and growth conditions sets a new founda-

tional baseline from which more complex plant science traits can be assessed for genotype-to-phenotype connections.

Acknowledgements

I want to express my heartfelt gratitude to Professor Guohui Lin and Professor R. Glen Uhrig for valuable guidance, thoughtful insight, support and encouragement. Without their guidance I could not complete this work. Their priceless insights and suggestions will always be guidance for my future career and life.

I would really like to thank Enrico Scarpella (University of Alberta) for the provision of all photoreceptor mutants. I appreciate the plant cultivation, monitoring, and data preparation work done by Christian Lummer and Sabine Scandola. The implementation of the camera system and image processing were both done in part by Sabine Scandola. Curtis Kennedy made contributions to dataset creation, image processing, and experimental machine learning algorithm analysis. Gemmary Chang contributed to the testing of deep learning models in some analyses. I have contributed to dataset preprocessing, postprocessing, dataset analysis, model implementation, parameter tuning, model output, and writing.

Lastly, I want to express my deepest gratitude to my family who are there in every aspect of my life. My mom and dad who supported me and encouraged me in all my life. My brother Tawseef who has been always a constant support for me, especially during my difficult times. Without them, I could not come this far in my life.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Evaluation, selection, and production of cultivars all heavily rely on genotype classification [80]. To meet the ever-increasing demands of the increasing human population, plant productivity must drastically improve by using resources more effectively. However, this depends on a thorough grasp of the genotype-phenotype link [64]. Effective plant phenotyping is practically important in a number of ways, including the identification of diseased plants, classification of different species or cultivars/genotypes and/or measurement of plant traits resulting from the interaction of a plant's genotypes with the environment [77], [38], [85]. Identification of plant genotypes is one of the necessary steps in plant variety selection and plant stress analysis. Plant phenotyping was initially carried out manually and hence was more prone to errors. Therefore, numerous image processing, computer vision, machine learning (ML), and deep learning (DL) based algorithms are utilised for plant trait estimation and classification tasks in order to remove human interaction and to increase overall accuracy [38].

In plant classification studies, time-series data provide important information such as seasonal trends, plant productivity over a season, plants' dynamic growth based on leaf area, etc [73], [49], [79]. Time-series data represents a critical data type for resolving trends in biological systems. Correspondingly, time series classification is crucial for the computer vision and machine learning communities since time series data are common in a variety of application

domains [5], [82], [52], [57]. From an agricultural perspective, extracting characteristics from time series data is critical in order to obtain relevant information on the state of growth and development [82], [52]. Studies demonstrating the power of computer vision have also looked at plant classification based on image processing, vesselness measure, images of overlapping leaves, as well as plant texture features [32], [36]. Further, research applying computer vision methods have succeeded in identifying plant diseases based on probabilistic classification of "healthy" vs "sick", images of diseased areas, signs of environmental stress, and identifying changes of electrical signals of plants due to environmental changes have been performed [71], [2], [84], [11].

In static plant analysis, computer vision and machine learning techniques have excelled, proving their capacity to learn plant growth patterns [59], [38], [85]. As plants are not static but constantly growing and developing on a daily basis, our research aims to construct a model for automatic genotype classification over a growth time-course using advanced machine learning and deep learning algorithms. Currently, supervised ML techniques have already been deployed for analyzing the biological properties of plants. Random Forest (RF), one of the supervised machine learning algorithms, is a non-parametric approach that has been applied to disease prediction, protein sequence selection, and gene selection [59], [18], [58], with plant biomass having also having been accurately predicted using RF image based data [11]. As well, support vector machine (SVM) has been applied to stress plant identification, neuroimage classification, plant image classification, biomass prediction [11], [25], [12]. Lastly, stacking multiple ML classifiers has demonstrated advantages for crop categorization estimation when compared to the use of a single classifier, suggesting that multiple classifiers in combination can lead to more robust classification outcomes [10] [86].

The plant sciences have also steadily incorporated DL methods, with experimentation using DL algorithms providing superior performance relative to conventional ML algorithms in classifying plants and in detecting various plant

diseases [16], [44], [4]. Convolutional Neural Networks (CNNs) have successfully classified plants [78], [45],[40], [76], [35], [4] and identified diseased plants [67], [3], [53]. For example, a CNN-based approach DenseNet-77 gave better accuracy than SVM and K-Nearest Neighbors (KNN) in detecting diseased plants [3]. CNN techniques have also been proven capable of differentiating plants according to species [40], [76], [35], [44], [83]. Recurrent Neural Networks (RNN) have also been successful in analyzing spatiotemporal data when paired with CNNs [27],[9], [47]. Long Short-Term Memory (LSTM; an RNN variation), has also been used for sequential data tasks due to its ability to capture long time-frame dependencies [31], while LSTM and Bidirectional-LSTM (the improved architecture of LSTM) approaches have successfully aided in assessing rice cultivation in southern Brazil [15]. Here, the authors compared their results with classic ML methods, including SVM, RF, k-Nearest Neighbors (k-NN), and Normal Bayes (NB). Based on the Densenet201 and bidirectional LSTM, a Densenet201-BLSTM model was proposed for classifying various genotypes based on time-series of plant images [80]. In the model plant *Arabidopsis thaliana*, a CNN-LSTM method proved most useful in classifying four accessions (Sf-2, Cvi, Landsberg (Ler-1), and Columbia (Col-0)) for plant growth differences and to categorize genotypes over plant development using single images over multiple days [77]. Alternatively, CNN with convolutional-LSTM (ConvLSTM) layers also demonstrated success when re-analyzing the same data [38].

Arabidopsis thaliana is a model plant species with extensive, well-characterized genetic resources for use in training ML and DL models [77], [38]. Critically, plant growth and development are highly dependent on sunlight, rendering our ability to detect genotype-to-phenotype differences fundamentally connected to light detection and core to agricultural applications. Light energy provides plants with a means to photosynthesize for growth [19] and a means to detect the environment for development [22]. Plants detect light signals, such as changes in light amplitude, color, spectra, and photoperiod, using a class of proteins called photoreceptors [24]. This enables plants to respond to changes

in their environment, such as seasonal transitions, day-night cycles [14], or shade from other plants [81]. There are four families of photoreceptors: Phytochromes (PHYA-PHYE), Cryptochromes (CRY1, CRY2, and CRY3), Phototropins (PHOT1 and PHOT2), the ZTL/FKF1/LKP2 group proteins, and lastly the UV-B resistance 8 (UVR8) family proteins [24], [46], [42]. Phytochromes absorb light in the red and far-red regions of the visible spectrum [54], [65] and regulate key developmental events such as seed germination, timing of flowering, size and shapes of plants and leaf movement [33], [75], [21], [74], [51], [66], [23], [39]. Cryptochromes detect blue and UV-A light [69]. They function during de-etiolation (the transition to the greening stage after plant germination; CRY1), in the photoperiodic control of flowering (CRY2), in the inhibition of the hypocotyl growth and in shade avoidance mechanism (CRY1 and CRY2) [28], [17], [1], [50]. Phototropins and ZTL/FKF1/LKP2 group proteins are sensitive to blue light. Phototropins control phototropism, light-induced stomatal opening, and chloroplast movement in response to changes in light intensity and direction [37]. Lastly, the ZTL/FKF1/LKP2 group proteins promote degradation or maintenance of circadian transcription factors, induce transitions in the day-to-dark transition [43], and are also involved in flowering, while the UVR8 family proteins absorb UV-B to signal harmful ultraviolet radiation [60]. While much has been resolved about how light activates plant photoreceptors, there are still gaps in our understanding of how these photoreceptors are connected to different elements of diel plant cell processes to affect phenotypic changes [70].

As plants are not static, but constantly growing and developing on a daily basis, our study aims to make genotype-to-phenotype connections using time-series growth data across multiple scales (intra- and inter-day data), genotypes, and growth conditions. To do this, we grew 17 different photoreceptor mutants under different light conditions to define the effects of twilight on a well-characterized population of photoreceptor deficient plants. To categorize these 17 genotypes based on their phenotypic responses under these different twilight conditions, we extensively tested a number of ML models, including: Support Vector Machine (SVM), Logistic Regression (LR), ensemble

stacking techniques (Random Forest (RF) and Boosting), alongside multiple DL techniques such as Fully Convolutional Network (FCN), Resnet, Encoder, Bi-LSTM, Conv2D, and ConvLSTM2D. We find that although conventional ML models were successful in categorizing genotypes under different twilight lengths, DL techniques perform much better, which we attribute to their ability to utilize multiple types of time-series data. In particular, our results demonstrate that while SVM maintains greater accuracy in classification tasks, the combined ConvLSTM2D model produced the best classification results for the various genotype classes of *Arabidopsis thaliana* across the different twilight conditions.

1.1 Thesis Contribution

The contribution of this study is as follows:

1. One of the major contribution of this study is to create the dataset in lab by using Raspberry Pi, and cameras to take picture for every 5 minuts of 14 days and to extract plant area measurement under different light conditions using PlantCV. The dataset contained numerous flaws, including missing values, outliers, and consistency issues. However, with patience and thorough careful measurements multiple times, we were able to reduce the likelihood of errors in our dataset. We also did some data manipulation, data preprocessing before using them for analyses.

2. We applied our machine learning classifiers by taking into consideration time-series growth data across multiple scales (intra- and inter-day data), genotypes, and growth conditions. Here, in case of multiple scales, we considered high frequency fluctuations like plant growth data per day as well as longer term trends or patterns like plant's growth over 14 days.

3. We proposed both traditional and deep learning models to evaluate their ability to differentiate 17 well-characterized photoreceptor deficient genotypes differing in their light detection capabilities grown under several different light conditions. We employed 0-min, 30-min, and 90-min light

conditions for plants growth and evaluate the models' performance by their ability to differentiate plants grown under various light conditions.

4. The optimal parameter settings for each of the conventional and deep learning models we provided for classifying plants were identified. To make the results easier to grasp, we cross-validated the results and displayed them in boxplots.

5. We demonstrated the performance of each models on time-series plant dataset and discussed the factors contributed to it. Finally, we suggested that the ConvLSTM2D model and SVM had the greatest performance for classifying plant genotypes.

6. We experimented and figured out which photoreceptor mutants of *Arabidopsis Thaliana* plants have more likely growing patterns and conditions like wild type plants so that it can help to identify which genes are responsible for the plant's growth and development independent of light conditions.

1.2 Thesis Outline

The rest of the thesis is organized as follows:

1. Chapter 2 provides detailed explanation of Material and Methods we used. Section 2.1.1 explains the used plant materials and the growth condition of the plants in lab. Section 2.1.2 explains the phenomic measurements containing the computers, cameras and other technological devices used to extract plant data. Section 2.1.3 explains dataset construction and section 2.1.3 explains the input data structure.

2. Section 2.2 provides detailed explanation of all the traditional and deep learning models we used for our experiment with model architecture, hyperparameter optimization, cross-validation, and model evaluation.

3. Chapter 3 provides detailed performance analysis of the models and discussion of the results.

4. Chapter 4 concludes this thesis with the summary of the proposed methods and results. This section also includes the future prospect of the thesis experiments.

Chapter 2

Literature Review

Across the plant sciences, computer vision and machine learning algorithms have demonstrated their ability to understand plant growth patterns. With the steady adoption of DL techniques, experiments have demonstrated that DL algorithms often perform better than traditional ML algorithms at classifying plants and identifying different plant traits. In this chapter, I discuss research relating to deployment of both the conventional machine learning models and deep learning models in the plant sciences.

The authors [63] proposed a statistical framework for the analysis of phenomics data by integrating DM and ML techniques. The most popular supervised machine learning methods are utilized to classify and predict plant health (stress/non-stress), including Linear Discriminant Analysis (LDA), Random Forest (RF), Support Vector Machine with linear (SVM-l), and radial basis (SVM-r) kernels. They looked at several datasets that represented real and simulated plant phenotypes. They studied the performance of each ML method (classifiers) and investigated how performance varied with the chosen number of attributes. The classification accuracy for 20%, 30%, 40%, and 50% rank features was calculated for simulated data. All offered about the same level of classification accuracy as non-rank all features. Here, rank features have been lowered by up to 50% and produced excellent results (98%). On the other hand, for real data, the SVM model's prediction accuracy averages around 97%. According to the study, when the features were chosen using the suggested methods, there was no appreciable difference in classification accu-

racy among the investigated ML approaches, except for Random Forest (RF). This study demonstrated that issues with using ML methodologies to analyze phenotype data might be resolved by combining the DM and ML methods for trait identification and classification, respectively.

The paper [4] presents a comparative study of traditional image processing and deep learning techniques for plant recognition. In this study, two methods - the conventional method and the deep learning approach are used to identify plant species. Hu moments (shape features), Haralick texture, local binary pattern (LBP) (texture features), and color channel statistics are used in the classic method to extract features (color features). Several classifiers (linear discriminant analysis, logistic regression, classification and regression tree, naive Bayes, k-nearest neighbor, random forest, and bagging classifier) are used to categorize the retrieved features. Also, various deep learning architectures are evaluated in the context of identifying plant species. One real-time dataset (Leaf12) and three standard datasets (Folio, Swedish leaf, and Flavia) are used. With the Leaf12 dataset, it was found that using the conventional method, the feature vector produced by combining color channel statistics, LBP, Hu, and Haralick with the Random Forest classifier produced a plant recognition accuracy (rank-1) of 82.38%. For the Leaf12 dataset, the accuracy of the VGG 16 (CNN architecture) with logistic regression was 97.14%. For the Folio, Flavia, and Swedish leaf datasets, accuracy of 96.53%, 96.25%, and 99.41% were achieved utilizing the VGG 19 CNN architecture and logistic regression as the classifier. VGG (Very Large Convolutional Neural Network) CNN models were found to have a greater accuracy rate than conventional techniques.

The paper [3] presents a new approach to detecting and classifying plant diseases using deep learning techniques. The authors proposed a DenseNet-77-based CenterNet model. The proposed method involves the use of a convolutional neural network (CNN) model that takes images of plant leaves as input and outputs a probability distribution over different disease categories.

The authors evaluated their approach on the PlantVillage dataset containing 54,306 images of plant leaves, with 12 healthy and 26 diseased plant classes of 14 species of plants. The samples for all 14 species of the crop including Tomato, Potato, Apple, Grape, etc., are downloaded from the Plantvillage dataset. The samples in the Plantvillage database are varied in the aspect of changes in angle, size, color, light, and the presence of blurring, noise, etc., which makes it a diverse database for plant disease recognition. They also compared their method with several existing approaches Inception-v4, VGG-16, and ResNet-50. The results demonstrated its superior performance in terms of accuracy, precision, and recall. They also compared their approach with conventional ML models like SVM and KNN and outperformed them in evaluation metrics. The performance showed the dominance of deep learning models over conventional ML models.

This study [17] suggests AyurLeaf, a Convolutional Neural Network (CNN) model based on Deep Learning, to categorize medicinal plants using leaf attributes including shape, size, color, and texture. The authors developed a dataset consisting of 2400 images of medicinal plant leaves of more than 30 leaves from 40 different plant species commonly used in Ayurveda, a traditional system of medicine in India, and used it to train a Convolutional Neural Network (CNN) model for plant classification. To efficiently extract features from the dataset, a deep neural network modeled after Alexnet is used. Ultimately, Softmax and SVM classifiers are used to perform the classification. For the AyurLeaf dataset, the proposed model, after five cross-validations, had a classification accuracy of 96.76%. The paper presents a promising approach for plant classification using deep learning techniques and demonstrates its potential for practical applications in the field of medicine.

The paper [36] presents a comprehensive review of the use of Convolutional Neural Networks (CNNs) for image-based high-throughput plant phenotyping. Plant phenotyping refers to the measurement of plant traits such as growth rate, leaf shape, and disease resistance, among others, in a high-throughput

manner using imaging techniques. The authors first provide an overview of the challenges associated with traditional plant phenotyping methods and the potential advantages of image-based high-throughput phenotyping using CNNs. They then describe the different CNN architectures used for plant phenotyping, including classic architectures such as AlexNet and VGG, and more recent ones such as ResNet and DenseNet. They also discuss the potential challenges and future directions for the use of CNNs in plant phenotyping. Overall, the paper highlights the potential of these techniques for advancing the field of plant biology and agriculture.

The paper [81] "Leaf segmentation and classification with a complicated background using deep learning" by Yang et al. presents a deep learning approach to segment and classify plant leaves in images with complex backgrounds. Images of leaves with many targets and a complex background are segmented and classified using the Mask R-CNN model and the VGG16 model. For training and testing the model, more than 4,000 images were employed. It is advised to conduct additional research using other deep learning algorithms and more data, as this could produce superior results. Together with algorithm improvement, improving image quality with better hardware can also result in improved performance. The proposed approach achieved an average segmentation accuracy of 96.4% and an average classification accuracy of 91.3% on the test set, outperforming other state-of-the-art approaches. The authors also conducted experiments to evaluate the impact of the number of training images and the complexity of the background on the performance of the proposed method and found that the proposed approach is robust to variations in these factors. Overall, the study demonstrates the potential of deep learning methods for accurate and efficient segmentation and classification of plant leaves in complex images, which could benefit various applications in plant phenotyping and agriculture.

The paper [41] proposes a deep learning approach to classify crop types using both spatial and temporal features extracted from multi-temporal remote

sensing images. The authors used the Multi-Source Land Imaging Time Series (MuSLIT) dataset, which includes multi-temporal images of six crop types in four regions of China. The proposed approach involves a two-stage process. In the first stage, a 2D Convolutional Neural Network (CNN) is used to extract spatial features from each image. In the second stage, a Bidirectional Long Short-Term Memory (BiLSTM) network is used to learn the temporal dependencies between the extracted features from different time steps. The output of the BiLSTM is then fed into a fully connected layer for crop type classification. The authors compared their approach with other deep learning methods and traditional machine learning classifiers. Their proposed approach achieved the highest accuracy in crop classification, outperforming other deep learning methods and traditional machine learning classifiers. The results demonstrate that the proposed approach effectively captures spatiotemporal features in crop classification and has the potential to be applied in precision agriculture and crop management.

In this study [48], a brand-new dataset called Urban Planter for classifying plant species was introduced. It has 1500 photos divided into 15 categories. The research team took extra care to compile and annotate the dataset. Deep neural network techniques such as VGGNet (VGG16, VGG19), DenseNet, MobileNet, Inception-v3, and Inception-ResNet-v2 were investigated by the authors (fine-tuned and updated). The outcomes demonstrated that DNN models perform remarkably well for genotype classification of plants.

The paper [83] presents a deep learning-based approach for the automatic anatomization of plant roots using a combination of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. The authors aimed to develop a system that can accurately segment and analyze root images, which is an important step toward understanding the genetic and environmental factors that affect root growth and development. The dataset contains 4000 images of plates with each plate having five plants of Arabidopsis Thaliana, a widely used model plant species in biological research, to train

and evaluate their model. They divided the dataset into training, validation, and test sets and used data augmentation techniques to increase the size and diversity of the training data. The authors developed a CNN-LSTM model that consists of two parts: a CNN that processes the root images to generate a feature map, and an LSTM that analyzes the temporal dependencies between adjacent frames in a sequence of images to improve the accuracy of root segmentation. The authors used a combination of binary cross-entropy loss and intersection-over-union loss as the objective function for training the model. The authors evaluated the performance of their approach using several metrics, including pixel accuracy, intersection-over-union score, and root length measurement error. They compared their approach with several existing methods like VGG, FCN, ResNet, UNet, and FastFCN for root anatomization and demonstrated that their approach achieved superior performance on all metrics.

The paper [75] presents a deep learning-based approach for the classification of plant phenotypes and genotypes using temporal data. The authors aimed to develop a system that can accurately predict the phenotype and genotype of a plant based on its temporal development, which is important for understanding the genetic and environmental factors that affect plant growth and development. In this research, the authors suggested a CNN-LSTM framework for classifying different genotypes of plants. Instead of creating features by hand, they used deep CNNs to automatically develop joint features and classifiers. The authors discussed the issue of handcrafted features and used deep learning to solve it (CNN). In addition, the growth of the plants and their dynamic behaviors are studied as significant discriminative characteristics for accession categorization using the potential of LSTMs. They created the dataset by gathering a collection of time-series image sequences of four different Arabidopsis accessions (Sf-2, Cvi, Landsberg (Ler-1), and Columbia (Col-0)) top-view images that were taken under comparable imaging settings and may be utilized as a standard reference point by specialists in the field. The proposed framework for genotype classification consists of a deep CNN

visual descriptor (using a CNN) and feature extractor with an LSTM model that can recognize and synthesize temporal dynamics in an image sequence as well as texture changes. The performance of their proposed deep phenotyping system (CNN + LSTM) was compared to other baseline methods like using handcrafted features and SVM as a classifier, adding the LSTM to consider temporal information, CNN without temporal information, and using CRF instead of LSTM to compare their performance. CNN+LSTM outperformed all other methods and proved the potential in predicting the crop yield of the plants as well as their health in the future.

The paper [39] proposes a spatiotemporal deep neural network approach for the classification of Arabidopsis plant accessions using image sequences. The authors aimed to develop a system that can accurately predict the accession of a plant based on its spatiotemporal features, which can provide valuable insights into the genetic and environmental factors that influence plant growth and development. Using a temporal series of RGB images, the authors focused on the accession classification of four different accessions of Arabidopsis plants: Columbia (Col- 0), Cape Verde Islands (Cvi), Landsberg (Ler-1), and San Feliu (Sf-2). The overwhelming resemblance in appearance across the four accessions of the Arabidopsis plant pictures is a significant challenge in this undertaking. They introduced three deep neural network architectures, including the 3-dimensional (3-D) CNN, CNN with convolutional LSTM (ConvLSTM) layers, and vision transformer, which classify plant accessions using both temporal and spatial features. A 3-D CNN model, which considers the input image sequence as a 3-D cube, makes up the first network. The number of images in the sequence is handled as the depth parameter in addition to the height and width of individual image. CNN and RNN make up the second network. To determine spatiotemporal characteristics, they replaced the LSTM layer in the network with a ConvLSTM layer. The first two approaches rely on convolutions, but the third method—vision transformer—uses a self-attention mechanism and is not dependent on convolutions. The simultaneous input of image patches into the vision transformer network speeds up model training.

The authors evaluated the performance of their approach using several metrics, including accuracy and F1 score. They compared their approach with the existing method [75] for plant accession classification and demonstrated that their approach achieved superior performance on all metrics. The authors also analyzed the spatiotemporal features learned by their model and showed that it was able to capture the important characteristics of the Arabidopsis plant accessions, such as leaf shape and size.

The paper [78] proposes a high-accuracy genotype classification approach using time series imagery for crop plants. The authors aimed to develop a system that can accurately predict the genotype of a crop plant based on time series imagery, which can provide valuable insights into the genetic and environmental factors that influence plant growth and development. A Densenet201-BLSTM model is proposed for classifying various genotypes based on time series of plant images. Densenet201 model is based on the Densenet201 and bi-directional Long Short-Term Memory model (bi-directional LSTM). This study offered an integrated dataset containing segmented plant images of growth and development that included 4 genotypes of Arabidopsis thaliana and 39 genotypes of panicoid grain crops, including maize, sorghum, and millet. The authors developed a high-accuracy genotype classification approach using a combination of a deep convolutional neural network (CNN) and a long short-term memory (LSTM) network. CNN was used to extract features from each image in the time series, and the bi-LSTM was used to model the temporal dependencies between the images. By bi-directionally capturing the dynamic behaviors of plant growth and development as well as important phenotypes, the proposed Densenet201-BLSTM represents the complex relationship between phenotypes and genotypes. The authors evaluated the performance of their approach using several metrics, including accuracy and F1 score. They compared their approach with several existing methods like Alexnet-LSTM, VGG16-LSTM, etc., for genotype classification and demonstrated that their approach achieved superior performance on all metrics. The proposed DenseNet201-BLSTM model obtains a genotype classification ac-

curacy of 98.31% on the test dataset. The proposed research was the first attempt undertaken to classify the genotypes of panicoid grain crops.

Overall, the contribution of machine learning and deep learning-based approaches in classification of plant genotypes is significant as it provides accurate and efficient methods for plant classification. These approaches can help in identifying the genetic traits responsible for plant growth and yield, which can aid in the development of new plant varieties with improved yield and disease resistance. Furthermore, these methods can be used to identify plant phenotypes that are tolerant to environmental stresses, such as drought and high temperatures, which could benefit various applications in plant phenotyping and agriculture.

Chapter 3

Materials and Methods

3.1 Materials and Methods

3.1.1 Plant Material and Growth Conditions

Experiments were performed using *Arabidopsis thaliana* plants, ecotypes Columbia (Col-0) and Nossen (No-0). Photoreceptor mutants (*phy A, phy B, phy C, phy D, phy E, phot 1, phot 2, phot1 phot2, cry1, cry2, cry1 cry2, cry3, ds-16, fkf1, lkp2, and ztl*) (Figure 3.1) were obtained from Dr. Enrico Scarpella (University of Alberta, Edmonton, Alberta). Seeds were sterilized in an air-tight container filled with chlorine gas for 24 hours. Chlorine gas was made by adding 3 mL of hydrochloric acid to 75 mL of bleach. The seeds were then placed on a $\frac{1}{2}$ Murashige and Skoog media containing 7 g/L of agar and 1% sucrose at pH 5.8 (KOH). After 3 days of stratification at $4°C$ in the dark, the seeds were exposed to light treatment for a week before being transferred to soil (Sungro, Sunshine Mix® 1). Growth chambers were equipped with a programmable Perihelion LED fixture (G2V Optics Inc.) and lined with Reflectix® to ensure a good light diffusion. Plants were grown under a 12h light and 12h dark photoperiod with a temperature of $21°C$ during the day and $19°C$ at night. Light treatments consist of different dawn and dusk ramp conditions for a given spectrum (Figure 3.1). Six LED types (Cree LED XPE, XPE2 XPG families) were used in the fixtures, with characteristic wavelengths of 444 nm, 630 nm, 663 nm, 737 nm, 3000 K white, and 6500 K white.

3.1.2 Phenomics measurements

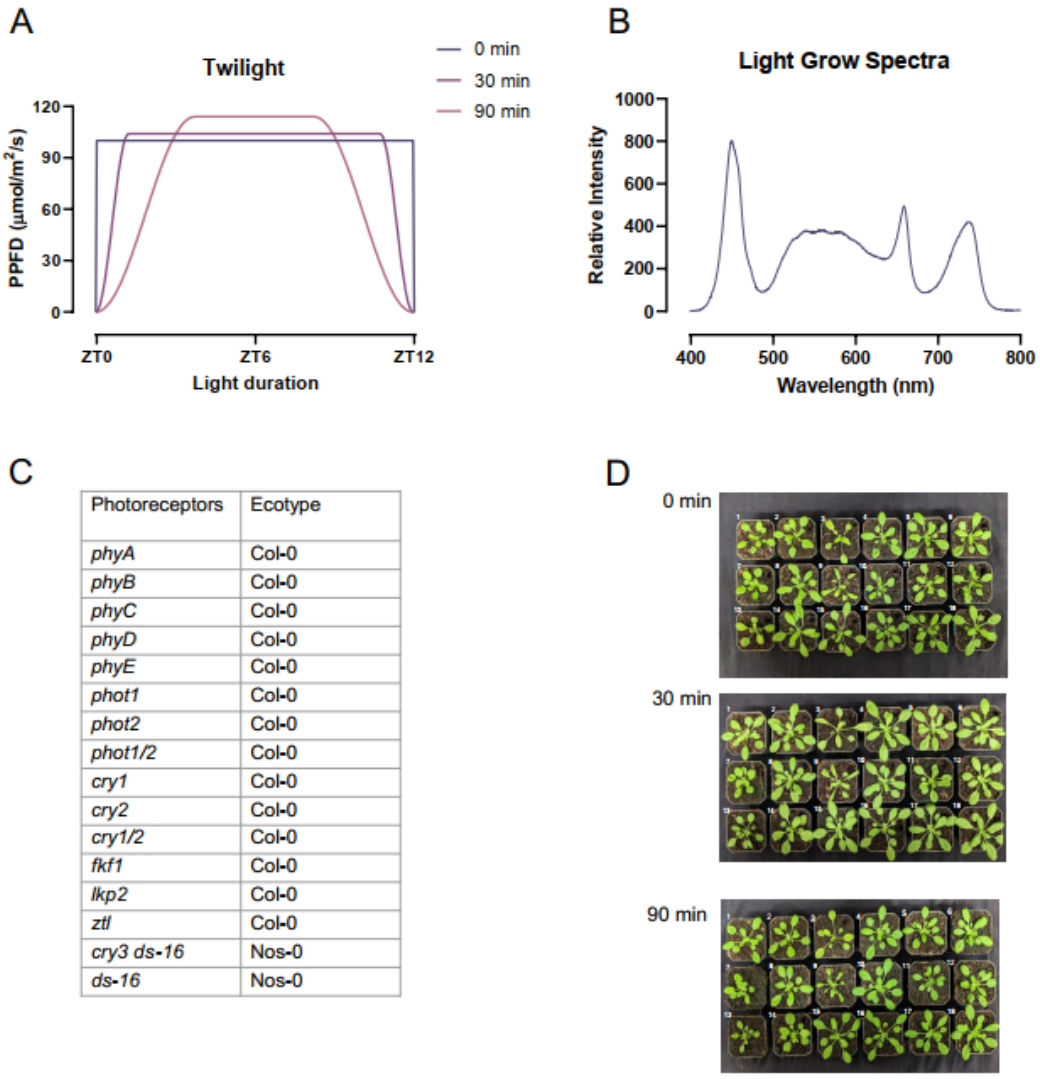
Each chamber was equipped with two Raspberry Pi 3 B+ and an ArduCam Noir Camera (OV5647 1080p) with a motorized IR-CUT filter and two infrared LEDs. Pictures were taken every 5 min over 14 days and were used to extract plant area and perimeter measurement using PlantCV, an open-source Python package with computer vision tools for plant phenotyping as previously described [26].

3.1.3 Dataset Construction

To create our dataset, images were processed using PlantCV [26]. The workflow consists of three core steps. First, image preprocessing was required to undistort the fisheye effect of the ArduCam Noir Camera. Images also receive exposure correction using a defined white spot region of interest. The second core step was image segmentation, in which pixels are classified into 'plant' and 'background' using an HSV color threshold. Segmented images were then filtered to remove small noises. The final core component of our PlantCV workflow was image analysis. Using the segmented images, measurements were taken of the plant dimensions and stored in the output file. The measurement of a size marker was also taken, which served as a constant. This allowed us to normalize the data across different cameras and account for slight variations in the distance between the plants and the camera. Additionally, as the size marker has a known size, it allows the measurements to be converted from pixels to SI (International System of Units) units.

3.1.4 Input Data Structure

Our dataset has 501 plants in total of 17 different genotypes of 14 days. Instead of feeding the entire 14 days of data at once, we divided each plant into different days, creating a dataset of 7014 plants of a single day. It helped to better understand the growth for each plant per day analysis as well as the increase in the total number of samples for training and testing the models. In a DL time-series classification model, input data arrays from the input



Photoreceptors	Ecotype
phyA	Col-0
phyB	Col-0
phyC	Col-0
phyD	Col-0
phyE	Col-0
phot1	Col-0
phot2	Col-0
phot1/2	Col-0
cry1	Col-0
cry2	Col-0
cry1/2	Col-0
fkf1	Col-0
lkp2	Col-0
ztl	Col-0
cry3 ds-16	Nos-0
ds-16	Nos-0

Figure 3.1: Light treatments and plant phenotype.
(A)Twilight ramps employed in the study. For the 0-minute square bracket ramp, lights turned on to full Photosynthetic Photon Flux Density (PPFD) or light intensity, while under the 30-minute and 90-minute conditions, the light intensity progressively increased to reach its maximum in 30-minute and 90-minute respectively. For all conditions, the plants received the exact amount of light of 4.32 DLI (mol/m2/d).
(B) Spectral composition of the light for all treatments. (C) Plant list and associated genotypes. (D) Pictures of representative photoreceptor-deficient plants grown under different twilight lengths. Plant position: 1 = *WT*, 2 = *phyA*, 3 = *phyB*, 4 = *phyC*, 5 = *phyD*, 6 = *phyE*, 7 = *phot 1*, 8 = *phot 2*, 9 = *phot1/2*, 10 = *WT*, 11 = *cry1*, 12 = *cry2*, 13 = *cry1/2*, 14 = *cry3*, 15 = *ds-16*, 16 = *fkf1*, 17 = *lkp2*, and 18 = *ztl*.

nodes are transmitted to the hidden layer of the network model, where the network model processes and learns the historical pattern before producing any prediction. The time-series of each chosen station variable was normalized to a similar range of values to guarantee that the proposed sequential neural model trains well and converges quickly. Additionally, the input data arrays at various sample rates produced adverse effects on the models' training and validation performances since the collected time-series observations were noisy and stochastic in nature. In our plant dataset, we found inherent biological variance in plant sizes with respect to area, even in plants of the same genotype. To increase the model's capacity for generalization, we used data normalization technique [56]. Here, we used the min-max normalization technique [13], where the initial data was transformed linearly using min-max normalization, also known as feature scaling. Using this method, we obtained all data scaled within the range (0, 1). That is, every feature's minimum and maximum values were each converted to a 0 and a 1, respectively, while all other values were converted to a decimal between 0 and 1, using the formula:

The formula: $X = \frac{(X_i - X_{min})}{(X_{max} - X_{min})}$

Where, where X_i, is the area value (> 0) at a given time for a particular plant i; X_{min} and X_{max} are the minimum and maximum area values per plant respectively; X is the normalized value for plant i at the given time.

3.2 Proposed Methods

We proposed various machine learning models for our research. We separated them into conventional machine learning models and deep learning models to classify different genotypes of *A. thaliana*, and we tested their performance throughout different dawn and dusk periods of the plants (Figure 3.2). We used hyperparameter tuning to find out the optimal settings of each models to generate the best classification accuracy. Moreover, we cross-validated all the results using 10-fold cross validation techniques. Finally, we evaluated our

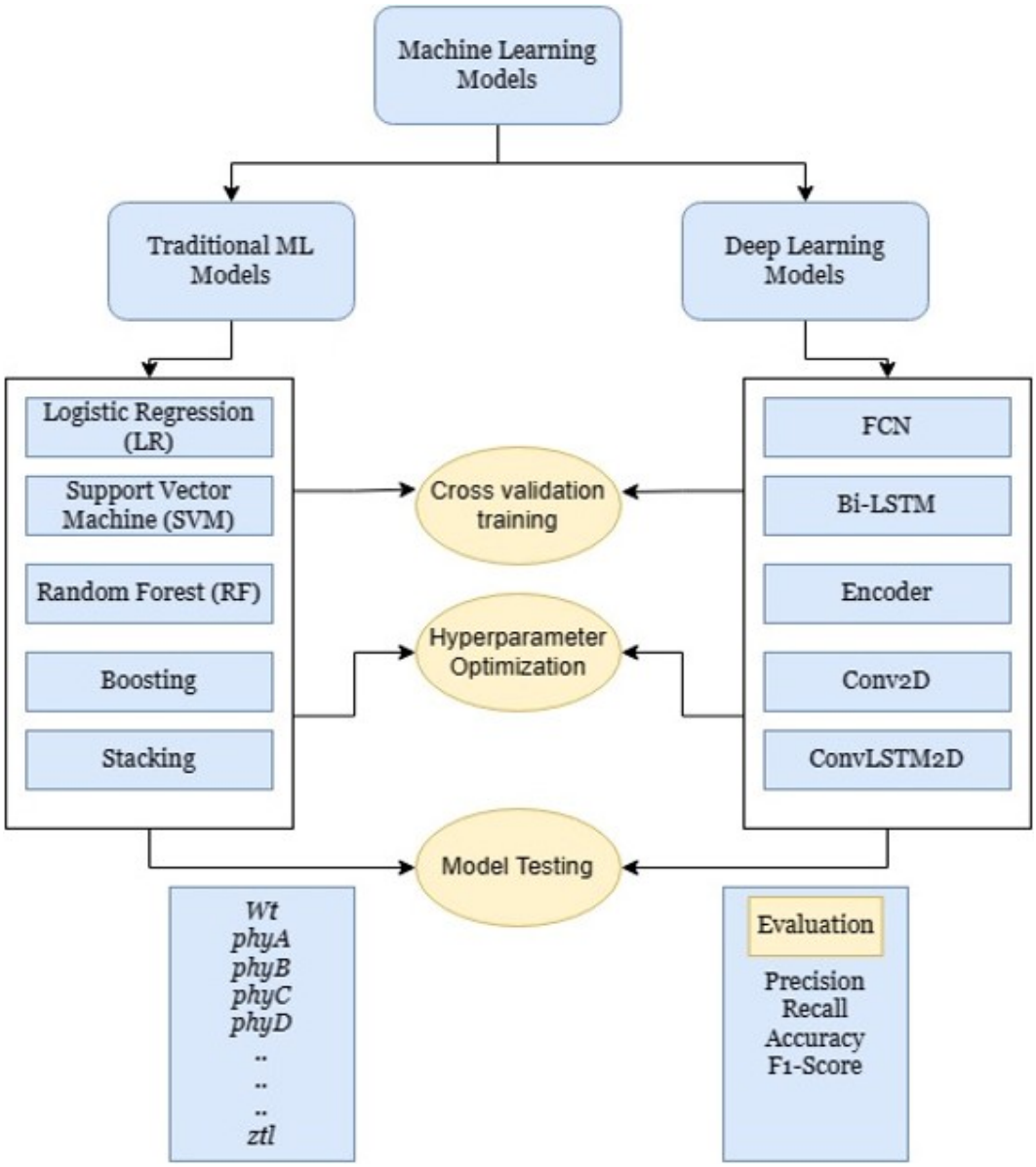


Figure 3.2: Proposed methodology for classification of plant genotypes.

models using precision, recall, f1-score, and accuracy metrics.

Support Vector Machine (SVM)

SVM is effective in time series because it can resolve issues with nonlinear regression estimation. Due to its successful use in classification and regression problems, it has been a popular subject of intense study. In order to make the input vectors linearly separable, the input vectors are mapped into a high-

dimensional feature space using SVMs. The kernel functions k(x, y) make it easy and efficient to do the non-linear mapping from the input vector space to the feature space (i.e., the so-called kernel trick). We do not need to take into account all the points but taking only a subset of points becomes beneficial as the SVM classifier is dependent ideally only on a subset of points while optimizing the distance between the nearest points of two classes (Margin). SVM accepts as input a set of data points in a vector space. In our case, a data point represents the plant area at a particular time point, and each dimension represents the total growth feature of the plant.SVM models work by finding the best separating (maximal margin) hyperplanes between the classes of training samples in the feature space. The margin is the separation between the nearest data points and a separating hyperplane. The hyperplane that provides the maximal possible margin is known as an optimal separating hyperplane, and these closest data points are known as support vectors. Finding a hyperplane can help with accurate data classification into different groups.

SVM models are created to predict or classify input data that belongs to two separate classes. However, SVMs can be utilized as multiclass classifiers by treating a K-class classification problem as K two-class problems. This categorization is referred to as one-vs-rest or one-vs all. The multi-class dataset is divided into various binary classification issues. Each binary classification problem is then fed into a binary classifier, and the best accurate model is used to make predictions. For multi-classification, SVM finds multiple hyperplanes that collectively best separate the data points into the designated number of classes (Figure 3.3). The strategy for handling multi-class classification can be set via the *"multi_class"* argument and can be set to *"ovr"* for the one-vs-rest strategy. In addition, SVM model requires 4 parameters (degree, C, ϵ, and γ) to be determined to provide accurate class predictions. C is the penalty parameter of the error term, ϵ defines a margin of tolerance where no penalty is given to errors, γ gives the curvature weight of the decision boundary, and *degree* parameter determines the degree of the polynomial used to find the hyperplane to split the data. Here, we used grid search method to find the op-

timal parameter setting of SVM for achieving the maximum accuracy with the minimum error. Ranges were set to [1, 10] at increment of 1.0 for C and [0.01, 0.001, 0.0001, 0.1 - 0.5] for ϵ with γ being fixed to be 0.5. The optimal values of *C, ϵ, and γ* are selected using 10-fold cross-validation repeated ten times to increase the reliability of the results. We tried SVM with several kernel types for our research and found that polynomial kernels with degree 4, C=1, $\epsilon = 0.1$, and $\gamma = 0.5$ delivered the best precision, recall, accuracy, and f1-score values. Propsoed SVM model works by mapping the time-series plant area data to a higher-dimensional feature space using a polynomial kernel function. Polynomial kernel function allows the model to capture non-linear relationships between the time-series plant area data and the genotype labels.Given a dataset, $P = \{(X_1, Y_1), ..., (X_v, Y_v)\}$, where $X_u \in X$ and $Y_u \in Y$, SVM try to solve the following equation:

$$\min_{G,b} \frac{1}{2}||G||^2 \tag{3.1}$$

subject to $y_u(G^t.x_u + b) \geq 1, u = 1, 2, ..., v$

The weight vector G and the bias term b are learned during the training of the SVM model. The goal of training is to find the values of G and b that minimize the classification error on the training data while maximizing the margin between the decision boundary and the training data points.

Kernel function is used for this particular problem which can't be solved using linear hyperplane. For degree-d polynomials, the polynomial kernel is defined as:

$K(x, y) = (x^{\mathsf{T}}y + c)^d$

where $c \geq 0$ is a free parameter that trades off the influence of higher-order vs lower-order parts in the polynomial, and x and y are vectors in the input space, that is, vectors of features derived from training or test samples. The kernel is said to as homogenous when $c = 0$. .

SVM model employed the normalized area values of plants for training and testing as described above.

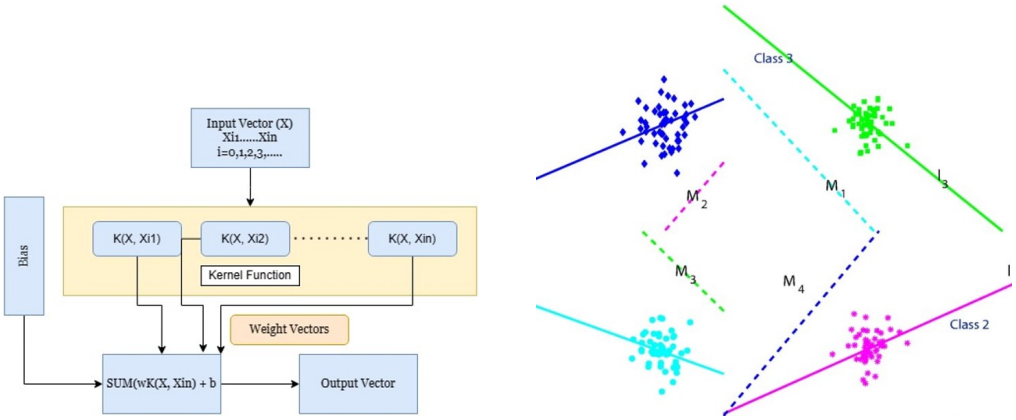


Figure 3.3: SVM multiclass classification with kernel function

Logistic Regression (LR)

LR works by estimating the probabilities of events, including determining a relationship between features and the probabilities of outcomes. Here, we used multinomial logistic regression to accommodate multi-class genotype classification problems. Here, for our study, the input to the LR is a total plant as independent variable and the dependent variable is the 17 genotype classes. The input plant area values with randomly initialized weight values added with bias term is passed to *sigmoid* function and mapped to a probability in a range between 0 to 1. Multinomial logistic regression alters the logistic regression model to directly support the prediction of multiple class labels. To more specifically forecast the likelihood that a given input example (i.e., a particular plant) falls under each accepted class label (*wt, phyA, phyB, and so on*) and the class label with highest likelihood value is selected. Here, for our study, both the training and testing phases of the logistic regression (LR) model employed the normalized area values of plants for training and testing as described above. We split the dataset into 10-fold for training and validating the model's performance and finally, we reported the maximum accuracy with the minimum cross-validation error. Performance of LR model greatly depends on hyper-parameter C, maximum number of iterations, penalty, and solver. C is the regularization parameter in the logistic regression model. It controls the trade-off between achieving a low training error and a low complexity model.

Another parameter *max_iteration* is the maximum number of iterations for the solver to converge. The solver iteratively refines the weights and bias of the model to minimize the loss function. Here, we used grid search method to find the optimal parameter setting of LR for achieving the maximum accuracy with the minimum error. Ranges were set to [0, 4] at increment of 0.2 for C and [*'l1', 'l2'*] for the regularization parameter, [*'newton-cg','lbfgs'*] as classifier solver. By grid search, we found the best parameter setting for logistic regression of *C=0.62, max_iteration = 200*, and *solver = "lbfgs"* with *l2* regularizer. As this is a multiclass problem, we set the parameter *"multi_class = multinomial", solver ("lbfgs")* handled multinomial loss, and 200 iterations were taken for solver for fast convergence of LR model.

Multinomial LR is based on Linear Regression, with the formula:

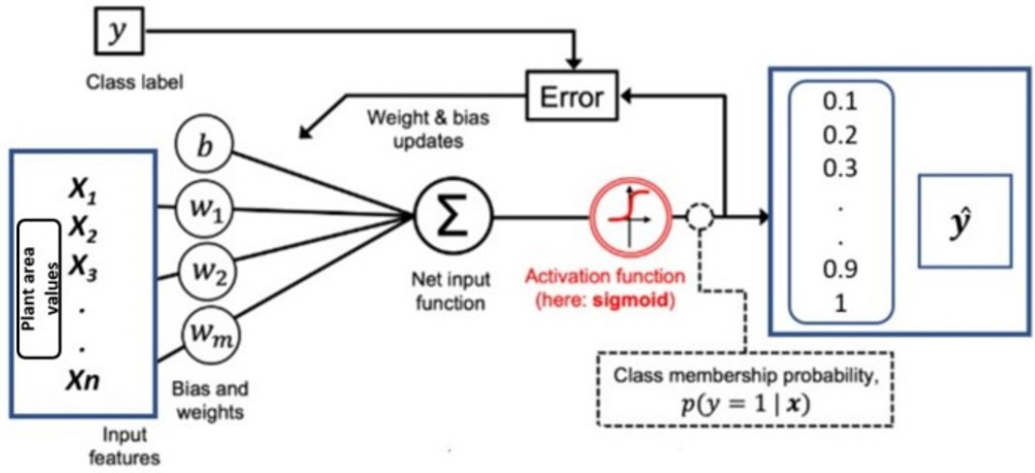


Figure 3.4: Logistic Regression (LR) model for time-series genotype classification of plants.

$y = mx + b$

Where y is our outcome variable that denotes the probability of a particular plant in a particular genotype, m is the curve slop that is the randomly initialized weight value, The maximum likelihood estimation(MLE) can be used to estimate the logistic function's parameter m. In MLE, the parameters that best match the joint probability of the independent variables x are sought after where x is the input features of plants, in our case, the area values of a plant. Here, b is the interception with the y-axis as a bias. As we have more than

one predictive variable our formula for classification tasks will look like this:
$f(x) = m_1x_1 + m_2x_2 + m_3x_3 + ... + m_nx_n + b$

where, n = total number of input features, $x_{1...n}$ are the input features (plant area values at particular time point), and $m_{1...n}$ are the weight variables assigned to the plant area values. Each input feature is multiplied by a weight, and the resulting products are summed together with the bias term to produce a score for each possible output class (Figure 3.4). The weights are learned during training of the model. The probability equations with multinomial logistic regression for a 17 category classification task would look like: $P_r(y = 17) = \frac{1}{z^{e^{f(x)}}}$
where z is the sum of the $e^{f(x)}$ for all classes in the model. P_r is the probability of a particular plant belong to a certain class.

Random Forest (RF)

Random Forest (RF) is an ensemble classification algorithm, consists of a group of tree-based classifiers $h(x, (\theta)_k$, where, (k = 1,2,..,), x is the input vector and $(\theta)_k$ are independent and identically distributed random vectors [7]. Every tree in the forest gives a unit vote, designating the most likely class label for each input (Figure 3.4). The algorithm achieves higher performance on high-dimensional data by doing an implicit feature selection using a small collection of "strong variables". For proposed RF model, split criterion - Gini index, which denotes importance or feature relevance, is used to depict the results of feature selection. The impurity of an attribute in relation to each class is measured by the Gini index. Gini impurity reveals how frequently a particular feature is chosen for a split and the magnitude of its total discriminative score for the particular classification problem.

Performance of RF highly depends on *"max_features", "n_estimators", "min_sample_leaf", and "oob_score"* parameters to provide accurate classification performance [61]. Here, *"n_estimators"* denotes the number of trees the algorithm builds before averaging the predictions, *"max_features"* denotes

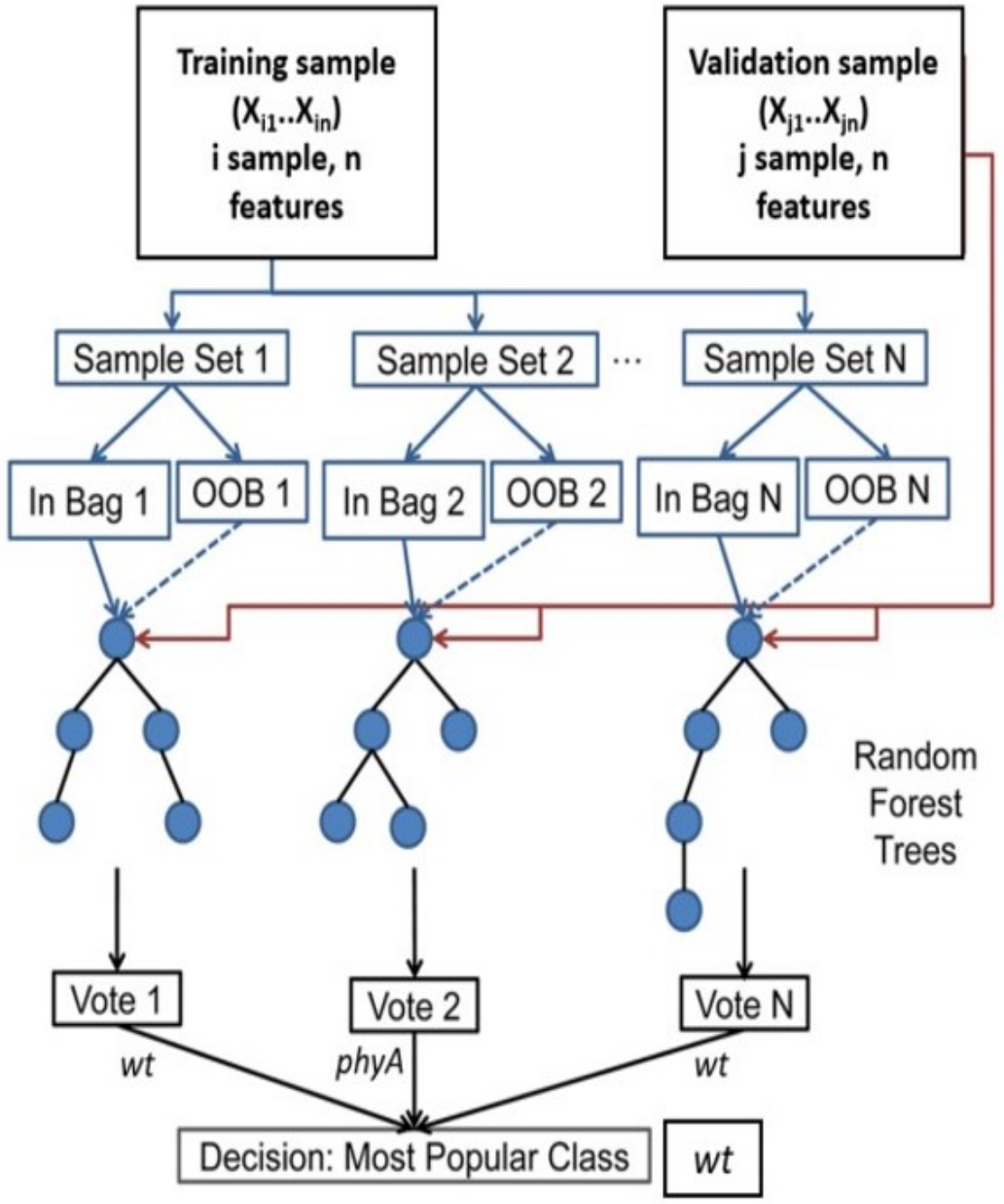


Figure 3.5: Random Forest (RF) model for time-series genotype classification of plants.

the maximum number of features random forest considers splitting a node, *min_sample_leaf* determines the minimum number of leaves required to split an internal node, and *"oob_score"* is a random forest cross-validation method. Here, we adjusted these parameters to run the RF model for our analyses, optimized by grid search parameter tuning. The primary parameters are the

number of predictors at each decision tree node split and the number of decision trees to run. For grid search, paramter ranges were set to [10, 2000] with an increment of 10 for *"n_estimator"*, [*'auto', 'sqrt'*] for *"max_features"*, [1, 30] with an increment of 2 for *"max_depth"*. Finally, we used *"max_feature = Auto"* to consider all the data points in an individual run, *"n_estimators = 200"* for better classification performance as well as faster running time, and *"oob_score = TRUE"* to use the RF self cross-validation method and can provide an estimate of the model performance without the need for a separate validation set. After trying multiple leaf sizes, we chose *"min_sample_leaf = 20"* to achieve the maximum accuracy with minimum error for validation data for RF model. RF model employed the normalized area values of plants for training and testing.

Boosting Ensemble

Machine learning models can be used individually or as part of an ensemble to fit data. A new model with greater effectiveness is created by the ensemble of simple individual models. Machine learning boosting can create an ensemble [88]. Boosting gives a prediction model in the form of weak prediction models, which are typically decision trees (DT). By learning straightforward decision rules derived from previous data, decision tree builds training model that may be used to predict the class or value of the target variable (training data). To predict a class label, decision trees begin at the tree's root and compare the root attribute's values with that of the attribute on the record. Then it follows the branch that corresponds to that value and goes on to the next node based on the comparison. Boosting creates an ensemble model by gradually integrating a number of weak decision trees. It gives each tree's output a weighted rating. Then, it increases the weight and input for inaccurate classifications from the initial decision tree. The boosting technique combines these several weak prediction rules into a single strong prediction rule after many cycles (Figure 3.6). While a strong classifier is arbitrarily well-correlated with the true classification, a weak classifier only modestly predicts the true classifi-

cation. Resampling data sets are also used in boosting to create classifiers, which are subsequently integrated by majority vote. For our study, we used GradientBoosting (GB) technique. By creating base learners (decision trees) consecutively, GB improves the loss function so that each base learner is always more effective than the preceding one. GB method produces accurate results at the beginning rather than fixing mistakes as they occur. Because of this, GB results in more precise results.

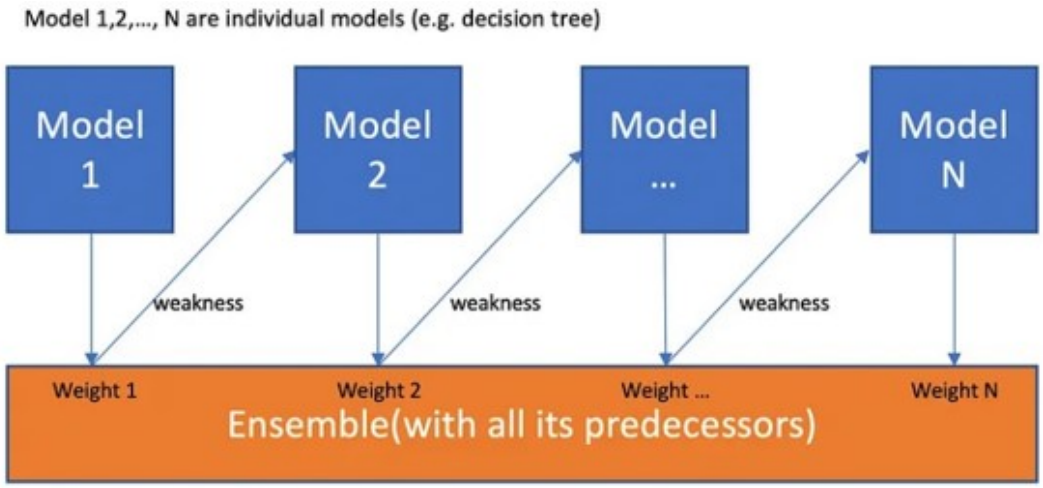


Figure 3.6: Gradient Boosting model for time-series genotype classification of plants.

For grid search, ranges were set to [10, 500] with an increment of 10 for *"n_estimators"*, [0.0001, 0.001, 0.01, 0.1, 1.0] for *"learning_rate"*, [0.5, 0.7, 1.0] for *"subsample"*, and [3, 7, 9] for *"max_depth"* parameter. Finally, We used *"n_estimator = 200"* denotes the number of decision trees that are added to the model sequentially in an effort to correct and improve upon the predictions made by prior trees, *"max_feature = Auto"* determines the maximum number of features decision trees considers splitting a node, for our case, we are considering all the features in a single run. We set *"learning_rate=0.01"* to update model weight and biases for convergence, *"loss=log_loss"* for better classification performance, *"max_depth = 7"* denoting max number of levels in each decision tree, and *"subsample = 0.5"*. We cross-validated the dataset 10 times as training and validation set and did this boosting process repeatedly 10 times.

Stacking

In stacking, predictions from various machine learning models are combined on the same dataset. The architecture of a stacking model consists of two or more base models [62], frequently referred to as level-0 models (in our study, RF, LR, and Boosting), and a meta-model, commonly known as level-1 model (i.e., SVC (*kernel = "poly"* for the study), that combines the predictions of the base models. The outputs from the base models that are utilized as input to the meta-model are probabilistic values or, in the case of classification, class labels (genotypes). The meta-model takes these input features and makes the final prediction for the plant genotype. We prepared the training dataset for the meta-model via k-fold cross-validation of the base models, where the out-of-fold predictions are used as the basis for the training dataset for the meta-model.

Convolutional Neural Network (CNN)

Our proposed CNN achitecture includes convolutional layers, max-pooling layers, and fully connected layers. Max-pooling layers are intended for feature selection, whereas convolution layers are intended for automatic feature detection. A convolution is applying and sliding a filter over the time series. For time series, the filters exhibit only one dimension (time) and seen as a generic non-linear transformation of a time series. This way, the kernel moves in one direction from the beginning of a time series towards its end, performing convolution. The elements of the kernel get multiplied by the corresponding elements of the time series that they cover at a given point. Then the results of the multiplication are added together and a nonlinear activation function is applied to the value. In our study, we have plant area values at each time point as input to the convolutinal layer. We have used only two convolutional layers where the first convolution has 128 filters with filter lengths of 7 and the second convolution has 256 filters with filter lengths of 5, respectively (Figure 3.7). These two convolutions are then input to pooling *AveragePooling1D* layer of *pool_size = 3* to reduce the dimnetionality. The output of pooling

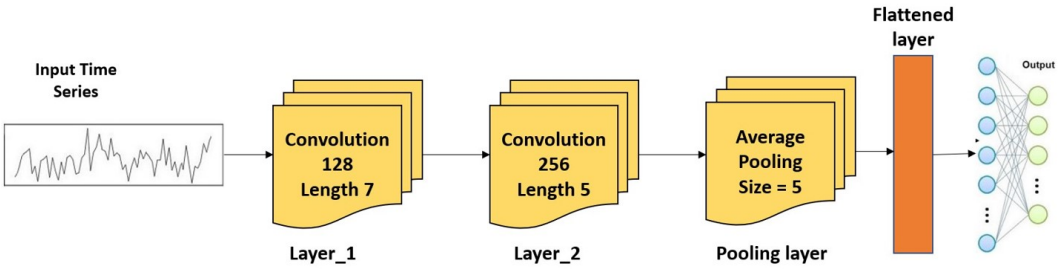


Figure 3.7: Convolutional Neural Network (CNN) model for time-series genotype classification of plants.

layer is fed to flatten layer to transforms multidimensional output and make it linear to pass it onto a dense layer. Finally, the ouput of flatten layer is fed to the output layer. The input, hidden, and dense layers used the *relu* activation function, but the model output dense layer used the *softmax* function having the exact number of neurons as genotype classes in the dataset.

Fully Convolutional Network (FCN)

We employed the architecture suggested by Wang et al. [82], which is made up of three convolutional blocks, each of which has three operations (a convolution, a batch normalization, and then a ReLU activation function [20]). Over

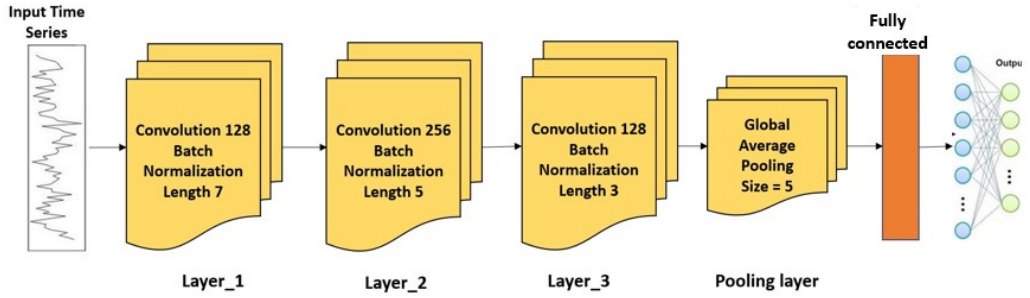


Figure 3.8: Fully Convolutional Network (FCN) model for time-series genotype classification of plants.

the complete time-series, which corresponds to the Global Average Pooling (GAP) layer, the third convolutional block's output is averaged. The output of the GAP layer is then fully coupled to a conventional softmax classifier with the same number of neurons as classes in the dataset [55]. The exact length of the time-series after the convolution is preserved by all convolutions having

a stride of 1 and zero padding. The first convolution has 128 filters with filter lengths of 7 and the second convolution layer has 256 filters with filter lengths of 5, respectively. These two convolutions are then input to the third and final convolutional layer, which has 128 filters with filter lengths of 3 (Figure 3.8).

Residual Networks (ResNet)

The deepest architecture for time-series classification is ResNet [82], which has 11 layers, the first 9 of which are convolutional layers and are followed by a Global Average Pooling (GAP) layer that averages the time-series over the time axis, minimizing the influence of vanishing gradients [29]. Our ResNet architecture consisted of three residual blocks, a GAP layer, and a softmax classifier with the same number of neurons as the number of genotype classes in the dataset as its final component. Each residual block is made up of three convolutions, the output of which is added to the input of the residual block and fed to the following layer. With the ReLU activation function being preceded by a batch normalization procedure, the total number of filters for all convolutions is fixed at 64. The length of the filter in each residual block is set to 8, 5, and 3 for the first, second, and third convolutions, respectively. Figure 3.9 represents the ResNet architecture that we used in our study of genotype classification.

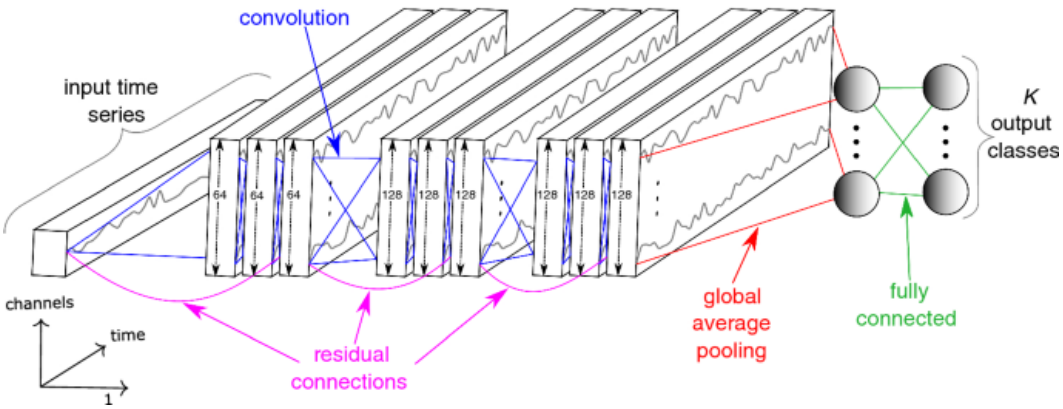


Figure 3.9: The Residual Network's architecture for time-series genotype classification of plants.

Long Short-Term Memory (LSTM)

LSTM architecture provides outcomes for sequential data workloads due to its capacity to capture long-time dependencies [31], [15]. LSTM architecture has a memory cell that maintains a current state over various sequential instances and non-linear dependencies that regulate information entering and exiting the cell. Bi-LSTM (an enhanced architecture of LSTM) models are typically more effective when managing contextual information since their output at a given time depends on both the prior and subsequent segments. To comprehend past and future information, the Bi-LSTM architecture contains two layers: forward and backward directions. Every series of events submitted to an LSTM is processed one time step at a time. A vector holding data about the current and prior time steps is passed from one time step to the next until it reaches the last one. However, the informational content of the vector will eventually be constrained by its fixed size. The information from the previous inputs runs the danger of being lost or diluted, especially for longer sequences. Here, we used the LSTM and Bi-LSTM (improved architecture of LSTM) with two hidden layers (256 LSTM units) and relu activation function for input and dense layers. For the output layer, we used softmax activation function having the same number of neurons as the number of genotype classes in the dataset. We trained the models with the following hyperparameters: (a) 250 epochs (after that no changes in validation loss and validation accuracy); (b) dropout rate of 0.3; (c) Adam optimizer [87] with a starting learning rate of 0.01; and (d) mini-batch size of 32. Additionally, categorical cross-entropy loss function was applied to monitor the validation loss.

Encoder

Our proposed encoder model is a standard convolutional network, with a convolution attention mechanism to summarize the time axis. We employed an encoder model with the following architecture (Figure 3.11). Our model has 3 convolution layers. The first convolution is made up of 128 filters with a length of 5, the second one is made up of 256 filters with a length of 11, and the third

one is made up of 512 filters with a length of 21. We used the Parametric Rectified Linear Unit (PReLU) activation function [30] that fed the output of each convolution's instance normalization operation. A dropout operation (with a rate of 0.2) and a final MaxPooling of length 2 are performed after PReLU's output. An attention mechanism that allows the network to learn which time-series (in the time domain) are essential for a specific classification was fed the third convolutional layer (Figure 3.10). Half of the 512 filters of last convolutional layer are input to the timewise softmax activation, which acts as an attention mechanism for the other half of the filters. Encoder represents a hybrid deep CNN [72], which is distinguished from FCN by replacing the GAP layer with an attention layer, minor modifications in convolutional layers, the PReLU activation function, the dropout regularization method, and the max pooling procedure. We applied 10-fold cross-validation on the datasets and validated the performance of encoder model by monitoring categorical cross-entropy loss of the validation set.

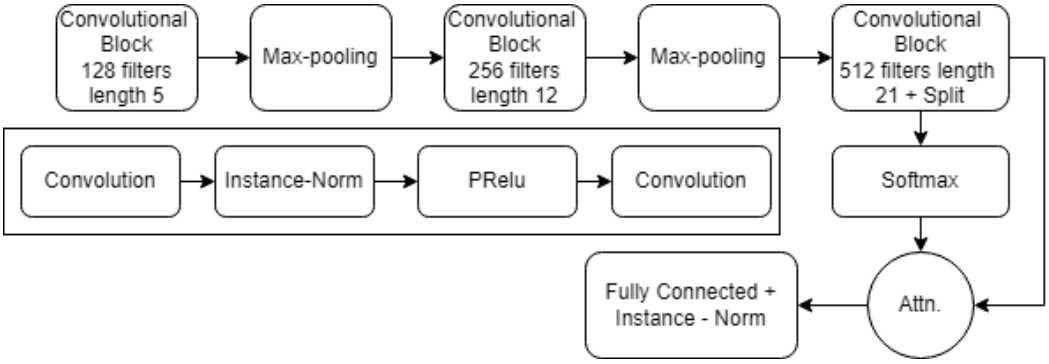


Figure 3.10: Encoder architecture for time-series genotype classification of plants.

Conv2D

For long-term non-linear input sequences, Conv2D model performs well. The 2-D MaxPooling layer and 2-D kernel size are features of the Conv2D model [56]. The Conv2D input shape for the input-layer was obtained using an *input_shape* argument [56] below: –

$$Input\_Shape\_Conv2D = (batch, timesteps, features) \tag{3.2}$$

We constructed the 7-layer Conv2D model to consist of an input layer, two 2-D MaxPooling layers, one convolutional hidden layer, one flatten layer, one dense layer, and finally output layer (2nd connected dense layer) respectively. The MaxPooling layer reduced the spatial dimensionality of the input sequence volume through downsampling approach. The flatten layer used the processed input sequence from the previous layer to narrow the featured sequence by wrapping it as a 1-D vector. Our Conv2D model had 128, 64, and 72 convolutional filters (layer neurons) for the input, hidden, and dense layers respectively, using a dropout of 0.3. The input, hidden, and dense layers were activated using the relu function, whereas the output layer was activated using the softmax function having the number of neurons as genotype classes in the dataset.

ConvLSTM2D

For the ConvLSTM2D model, the input data should maintain the following shape:

$$\text{Input\_Data\_ConvLSTM2D} = \text{X.reshape(samples, Input\_Shape\_ConvLSTM2D)}$$

$$= X.reshape(samples, batch, timesteps, features, channels) \tag{3.3}$$

For ConvLSTM2D, we created the 5-D ConvLSTM2D input arrays by using 3.3. The ConvLSTM2D model's system architecture (Figure 3.11) was then used to construct a seven-layer network, which consisted of the following layers: one input layer with the first batch normalization layer (instead of the first 2-D MaxPooling layer and dropout), one convolutional hidden layer with the second batch normalization layer (instead of the second 2-D MaxPooling layer and dropout rate), one flatten, one dense, and one output layer. The batch normalization layer shortened the learning time for the model and stabilized (regularized) the layer input arrays for a faster deep learning process [56]. The batch normalization layer was taken into consideration, which normalized an output data sequence from a prior connected layer (input or concealed layer). The flatten layer accepted the second batch normalization layer's processed

input sequence and wrapped it into a single 1-D vector (converting the input sequence from 5-D to 2-D output shape). The bundled input sequence was extracted and interpreted by the dense layer that was directly connected to the model flatten layer before being sent on to an output layer. The argument *"return_sequences"* was used. If *"return_sequences"* is true then ConvLSTM layer returns a sequence as a tensor comprising of sequence, filters, rows, and columns. The ConvLSTM layer delivers only the most recent output with filters, rows, and columns if the input *"return_sequences"* is false. Finally, the classification with the highest score was chosen using the output layer with the *softmax* activation function. The best potential combination was selected based on network performance after experimenting with various ConvLSTM layers and filter counts. After considering multiple hyperparameter settings, the following model hyperparameters were considered for best accuracy: padding was set to *"Same"* and the input, hidden, and dense layers all used the *relu* activation function, but the model output layer used the *softmax* function having the exact number of neurons as genotype classes in the dataset. We used the filter sizes for the input, hidden, and dense layers are 128, 256, and 1024 respectively. Our approach was to first pass each individual plant growth values with respect to area through the feature extractor (CNN) to produce a fixed-length vector representation. This fixed-length vector embodies the features of each individual plant. The outputs of CNN for the sequence of plant area values were then passed onto a sequence learning module (LSTM). At this stage, the LSTM attempted to classify the plants via analyzing the sequences of the features that were extracted from time-series area growth of plants and by considering their temporal variations.

3.2.1 k-fold cross-validation

To analyze all of the traditional machine learning and deep learning models, we used k(10) fold cross validation approaches rather than the standard train-test split. After completing cross-validation and averaging all the folds, we presented the results in terms of precision, recall, accuracy, and f1-score. Although we also tried 5-fold validation, 10 fold validation produced more

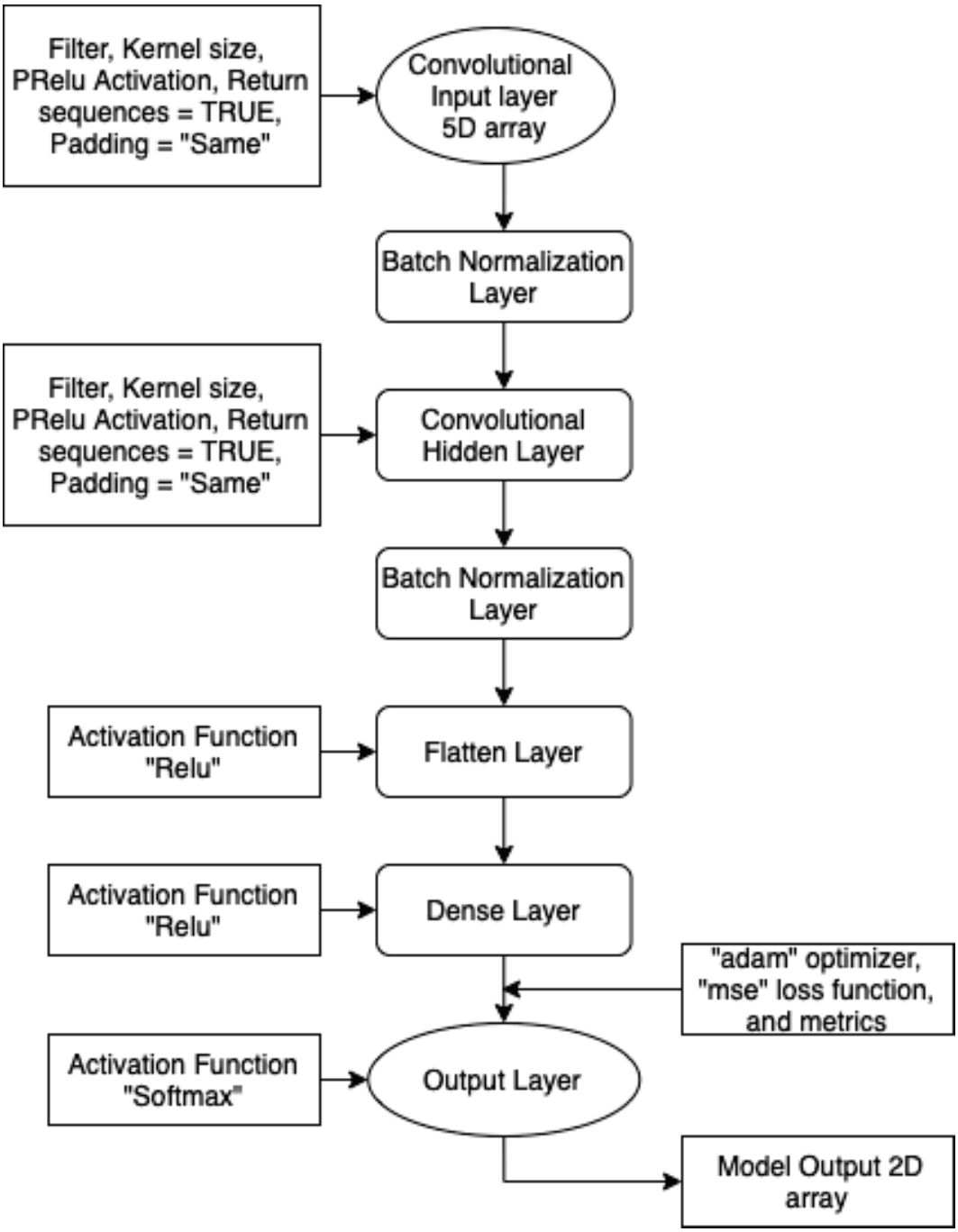


Figure 3.11: ConvLSTM2D architecture for time-series genotype classification of plants.

trustworthy and accurate results. We used k-fold (k = 10 in our analysis) cross-validation predictions as projected values for the training data. Here, we divided the training data d into k folds or groups. We started off with using k-1 fold as the test dataset and the remaining folds as the training dataset. We

trained the model on the training dataset and validated it on the test dataset and saved the validation score. These process repeat, but changing the value of k test dataset. Therefore, we selected k-1 as our test dataset for the initial round before moving on to k-2 for the subsequent round. By the time it was done, every fold in the model would have been validated, and the results would have been averaged to show how well the model worked. The predicted value that each plant in the training data acquired was the prediction while that plant was in the validation fold. To employ cross-validation procedures for the cross-validated prediction, all plants must be assigned to a validation fold exactly once.

After performing k-fold cross-validation, we do not need any additional statistical tests to determine which model is better than which model because k-fold cross-validation is itself a statistical technique that provides a reliable estimate of the model's performance. By repeating the training and testing process k times with different subsets of the data, we can obtain an estimate of the model's generalization performance that is less affected by the noise in the dataset than a single train-test split. Moreover, by computing the mean and standard deviation of the performance metric across the k folds, we can gain insights into the variability of the model's performance.

3.2.2 Hyperparameter optimization

The hyperparameter optimization is defined as a tuple of hyperparameters that produces an optimal algorithm that minimizes the predefined loss function (i.e., cross-entropy loss function in our study) on a held-out validation set of the training data. However, in our study, we carried out hyperparameter optimization by defining the subset of the hyperparameter space of an ML algorithm, then evaluating it by cross-validation using the training data. An exhaustive grid search approach was used. For deep learning models, we tested with different groups of parameters with every dataset and finally reported the parameters for which we acquired the best validation accuracy with the minimum validation loss. We set each deep learning model to 250 epochs. We employed the "EarlyStopping" approach [8] to stop iteration. Early stopping

technique prevent overfitting and improve the generalization performance of the model. It involves monitoring the performance of the model on a validation set during the training process and stopping the training when the performance on the validation set starts to degrade. We used "*patience* = 15". The "*patience*" parameter is a user-defined value that specifies the number of epochs to wait before stopping the training if the performance on the validation set does not improve. In our case, the *patience* is set to 15, the training process will continue for up to 15 epochs after the last time the validation loss improved. If the validation loss does not improve within the next 15 epochs, the training will stop.

3.2.3 Evaluation

To verify the effectiveness of the proposed approach for genotype classification, the evaluation metrics precision, recall, F1-score, and accuracy were used to compare the performance of the traditional and deep learning algorithms. Genotypes of plants were classified to verify the effectiveness of the proposed approach on each fold validation dataset. We have used the metrics precision, recall, F1-score, and accuracy to evaluate the performance of our models. Here, precision is defined as the actual correct prediction divided by total prediction made by model. For instance, our actual dataset has 130 *"wt"* plants, the model predicts 100 plants correcltly into *"wt"* class and 35 plants of other genotypes are misclassified as *"wt"*, then the precision score is .74 or 74%. On the other hand, recall measures the model's ability to detect actual samples and label them positive and is calculated as the ratio between the numbers of Positive samples correctly classified as Positive to the total number of Positive samples. So, the recall score is .77 or 77%. A substitute for accuracy metrics, F1-score is a machine learning model performance metric that equally weights Precision and Recall when assessing how accurate the model is. Precision, recall, F1-score, and accuracy are defined as the following:

$$Recall = \frac{TP}{TP + FN} \tag{3.4}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.5}$$

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3.6}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.7}$$

where TP, FP, and FN are the number of true positives, true negatives, false positives, and false negatives, respectively. True positives are data points that have been marked as positive and are in fact positive. For example, model predicts a plant as *"wt"*, which truly belongs to *"wt"*. False positives are data points with a positive label but a negative one. In our study, if a plant of other genotype misclassified as WT genotype. True negatives are data points that are truly negative despite being classified as negative. False negatives are data items with negative labels but positive values.

Chapter 4

Result Analysis and Discussion

In our study, we analyzed 17 different photoreceptor deficient plant lines grown under three different twilight conditions. We first evaluated four traditional ML algorithms and ensemble stacking, then tested additional six DL models to evaluate their performance. Here, we found that DL models consistently outperformed conventional machine learning models in terms of precision (P), recall (R), F1-Score (F1), and accuracy (Acc) (Tables 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, and 4.7). As we have a total of 17 genotypes, we created different datasets with 4, 6, 9, and 17 genotypes classes (by merging some closely related genotypes), respectively, to assess each model's performance, followed by the assessment of all algorithms on 3 separate datasets of 0-min, 30-min, and 90-min twilight growth conditions. We first evaluated traditional ML models for genotype classification in terms of precision, recall, F1-score, and accuracy (Tables 4.1, 4.2, and 4.3).

Here, we found that for each of the four datasets with 4, 6, 9, and 17 genotype classes, SVM with polynomial kernel outperformed all the other traditional ML models, achieving the highest precision, recall, F1-Score, and accuracy in all scenarios. Figure 4.1 and 4.2 represent the precision, recall, f1-score, and accuracy of the traditional machine learning models we used for our analyses. .

Under the 0-min twilight condition, SVM achieved an average accuracy of 73%, 64%, 54%, and 40%, respectively, at the presence of 4, 6, 9, and 17 genotype classes, respectively. The maximum accuracy among all growth

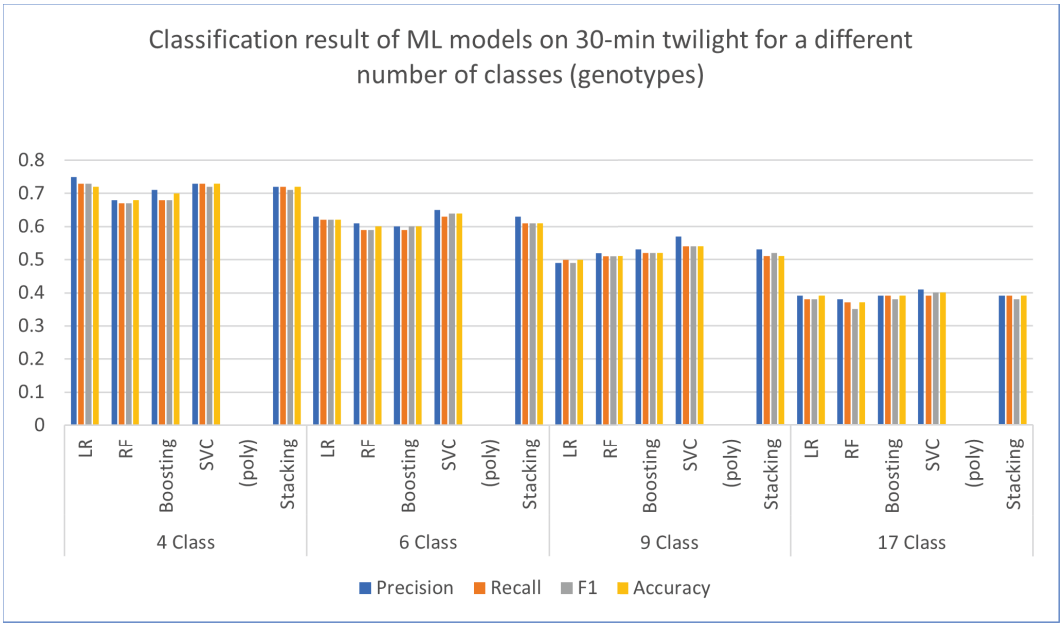


Figure 4.1: Classification result of ML models on 0-minute light conditions for a different number of classes (genotypes).

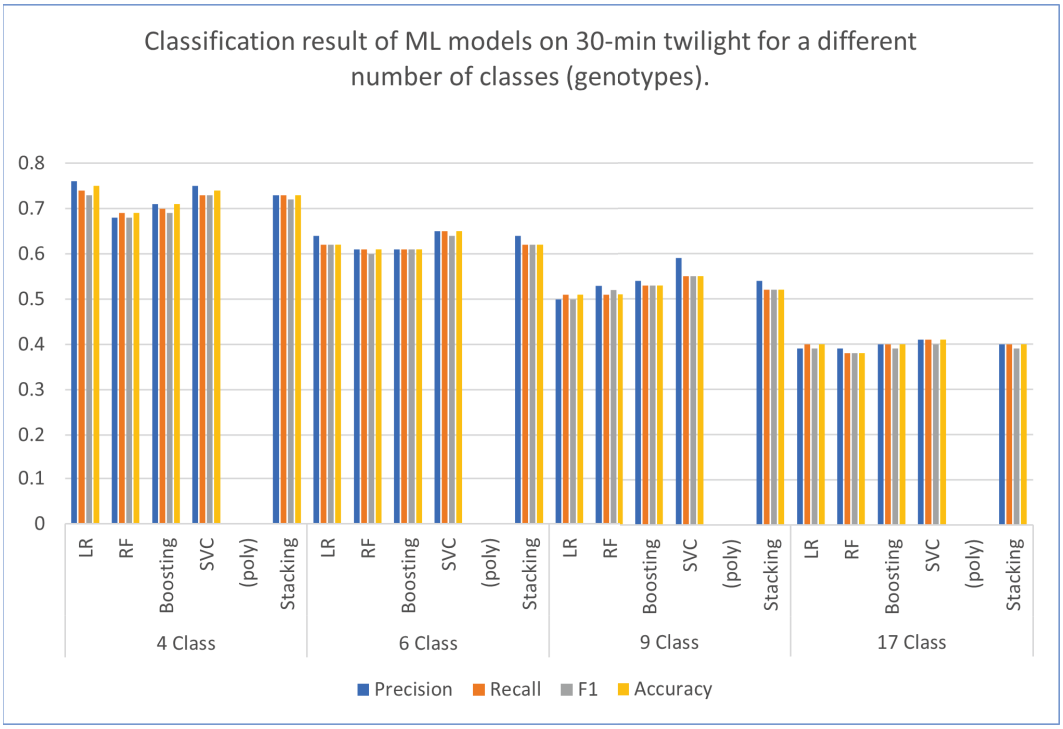


Figure 4.2: Classification result of ML models on 30-minute light conditions for a different number of classes (genotypes).

Table 4.1: Classification result of ML models on 0-min twilight for a different number of classes (genotypes)

No of Class	Models	P	R	F1	Acc
4 Class	LR	0.75	0.73	0.73	0.72
	RF	0.68	0.67	0.67	0.68
	Boosting	0.71	0.68	0.68	0.70
	SVC (poly)	**0.73**	**0.73**	**0.72**	**0.73**
	Stacking	0.72	0.72	0.71	0.72
6 Class	LR	0.63	0.62	0.62	0.62
	RF	0.61	0.59	0.59	0.60
	Boosting	0.60	0.59	0.60	0.60
	SVC (poly)	**0.65**	**0.63**	**0.64**	**0.64**
	Stacking	0.63	0.61	0.61	0.61
9 Class	LR	0.49	0.50	0.49	0.50
	RF	0.52	0.51	0.51	0.51
	Boosting	0.53	0.52	0.52	0.52
	SVC (poly)	**0.57**	**0.54**	**0.54**	**0.54**
	Stacking	0.53	0.51	0.52	0.51
17 Class	LR	0.39	0.38	0.38	0.39
	RF	0.38	0.37	0.35	0.37
	Boosting	0.39	0.39	0.38	0.39
	SVC (poly)	**0.41**	**0.39**	**0.40**	**0.40**
	Stacking	0.39	0.39	0.38	0.39

conditions was achieved by SVM for the 30-min twilight condition, where an average accuracy of 74%, 65%, 55%, and 41%, respectively, was obtained for 4, 6, 9, and 17 genotype classes. Observing that SVM with polynomial kernel performed exceptionally well in classifying time-series area data of plants, we remark that kernel selection and parameter search played a crucial role in its performance. It has also been reported that the effectiveness of SVM generalization (estimation accuracy) required a good setting of the kernel parameters, hyper-parameters C, and gamma [68]. SVM with polynomial kernel converged extremely quickly, handled nonlinear problems, affected the complexity of model selection, had fewer numerical challenges, and nonlinearly mapped samples into a higher dimensional space [6]. We found that the polynomial kernel, utilizing the grid search approach, performed the best across all kernels (*rbf, poly, linear, and sigmoid*).To test the effectiveness of SVC with various kernels, we used 30-minute twilight for nine distinct classes. For all of

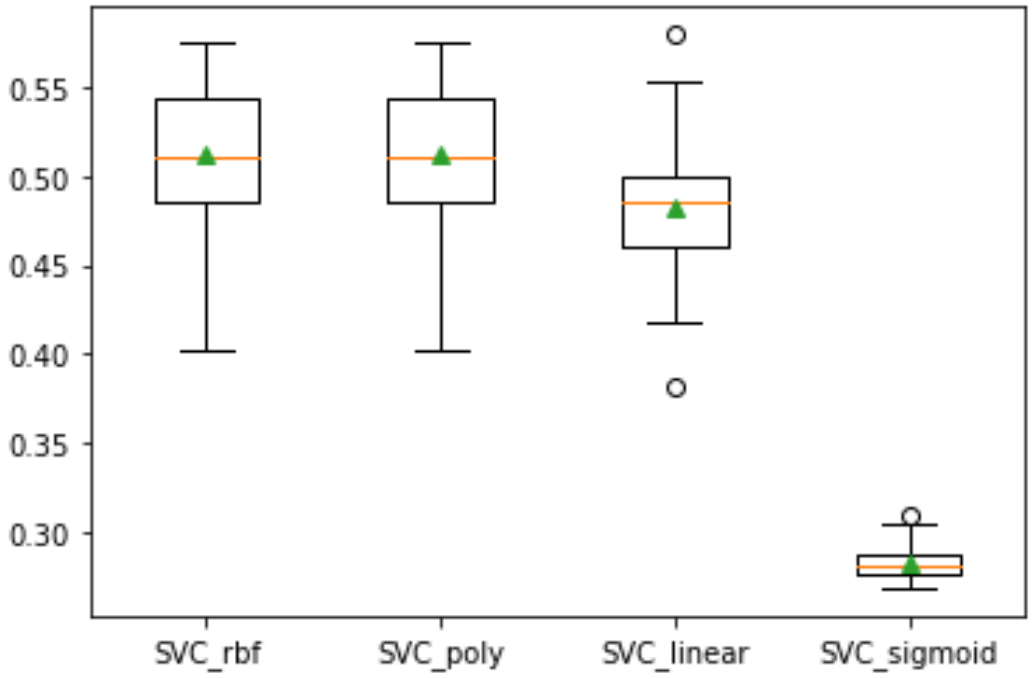


Figure 4.3: SVC performance on 30-min twilight condition for 9 different genotype class using different kernels.

the SVM kernels, we utilized 10-fold cross validation, and the average result was obtained and displayed as a boxplot (Figure 4.3). The boxplot makes it evident that SVC with a polynomial kernel performs better on average than all other results. SVM model is more robust to noise and polynomial kernel mapped out all the noises in higher dimensional feature space which ensure the results are more reliable.

In our experiment, Logistic Regression (LR) ranked the second-best for different twilight conditions. The parameter setup of LR model also played an important role in achieving higher prediction. We got the best parameter setting of LR model using grid search method.

Stacking also achieved satisfactory prediction results. Though it is designed to improve modeling performance, stacking is not always guaranteed for improvement in all cases. Given its lower complexity (i.e., it's easier to define, train, and maintain), the base model should be used instead of the stacking ensemble if it performs as well or better [8]. For instance, we experimented

Table 4.2: Classification result of ML models on 30-min twilight for a different number of classes (genotypes)

No of Class	Models	P	R	F1	Acc
4 Class	LR	0.76	0.74	0.73	0.75
	RF	0.68	0.69	0.68	0.69
	Boosting	0.71	0.70	0.69	0.71
	SVC (poly)	**0.75**	**0.73**	**0.73**	**0.74**
	Stacking	0.73	0.73	0.72	0.73
6 Class	LR	0.64	0.62	0.62	0.62
	RF	0.61	0.61	0.60	0.61
	Boosting	0.61	0.61	0.61	0.61
	SVC (poly)	**0.65**	**0.65**	**0.64**	**0.65**
	Stacking	0.64	0.62	0.62	0.62
9 Class	LR	0.50	0.51	0.50	0.51
	RF	0.53	0.51	0.52	0.51
	Boosting	0.54	0.53	0.53	0.53
	SVC (poly)	**0.59**	**0.55**	**0.55**	**0.55**
	Stacking	0.54	0.52	0.52	0.52
17 Class	LR	0.39	0.40	0.39	0.40
	RF	0.39	0.38	0.38	0.38
	Boosting	0.40	0.40	0.39	0.40
	SVC (poly)	**0.41**	**0.41**	**0.40**	**0.41**
	Stacking	0.40	0.40	0.39	0.40

with LR,RF, SVM, and Boosting as base models (Figure 4.4). However, we got better performance in SVM base model (0.53) than stacking (0.49). This suggests that the combination of multiple ML models can be more robust. In our experiment, Random Forest (RF) performed the average for almost all analyses. RF models have also been reported to produce strong results for multi-class classification problems; however, here we find them falling short of SVM for our multivariate time-series dataset. RF classifier is known for being more susceptible to noise than the other methods, and it has demonstrated robust performance with large datasets. RF classifier may not perform well for non-linear relationship between features or have complex dependencies.

The precision, recall, f1-score, and accuracy of the deep learning models we utilized for our investigations are shown in Figures 4.5 and 4.6. According to the figures, ConvLSTM2D model had the highest level of accuracy when predicting a variety of classes (genotypes) under various dawn and dusk twilight

Table 4.3: Classification result of ML models on 90-min twilight for a different number of classes (genotypes)

No of Class	Models	P	R	F1	Acc
4 Class	LR	0.74	0.74	0.74	0.74
	RF	0.68	0.68	0.68	0.68
	Boosting	0.69	0.68	0.68	0.69
	SVC (poly)	**0.73**	**0.73**	**0.73**	**0.73**
	Stacking	0.73	0.72	0.71	0.72
6 Class	LR	0.63	0.62	0.62	0.62
	RF	0.61	0.59	0.59	0.60
	Boosting	0.60	0.59	0.60	0.60
	SVC (poly)	**0.65**	**0.63**	**0.64**	**0.64**
	Stacking	0.63	0.61	0.61	0.61
9 Class	LR	0.49	0.50	0.49	0.50
	RF	0.52	0.51	0.51	0.51
	Boosting	0.53	0.52	0.52	0.52
	SVC (poly)	**0.57**	**0.54**	**0.54**	**0.54**
	Stacking	0.53	0.51	0.52	0.51
17 Class	LR	0.39	0.38	0.38	0.39
	RF	0.38	0.37	0.35	0.37
	Boosting	0.39	0.39	0.38	0.39
	SVC (poly)	**0.41**	**0.39**	**0.40**	**0.40**
	Stacking	0.39	0.39	0.38	0.39

conditions. Encoder model followed ConvLSTM2D with the second highest accuracy margin. For the DL models, we found that for each of the class sizes (4, 6, 9, and 17, respectively), ConvLSTM2D outperformed all the other models (Tables 4.4, 4.5, and 4.6), by achieving the highest precision (P), recall (R), F1-Score (F1), and accuracy (Acc) in all scenarios. Encoder model performed almost as well as ConvLSTM2D model, while LSTM model performed comparatively poorer than the other DL models. For all the models, we used 10-fold cross validation, took the accuracy score of each fold, averaged them, and finally decided based on the average score. In more detail, under the 0-min twilight light condition, ConvLSTM2D achieved an average accuracy of 79%, 66%, 56%, and 44% accuracy, respectively, for class sizes of 4, 6, 9, and 17. The maximum accuracy was reached by ConvLSTM2D under the 30-min twilight condition, with an average accuracy of 81%, 69%, 57%, and 45% for class sizes of 4, 6, 9, and 17, respectively.

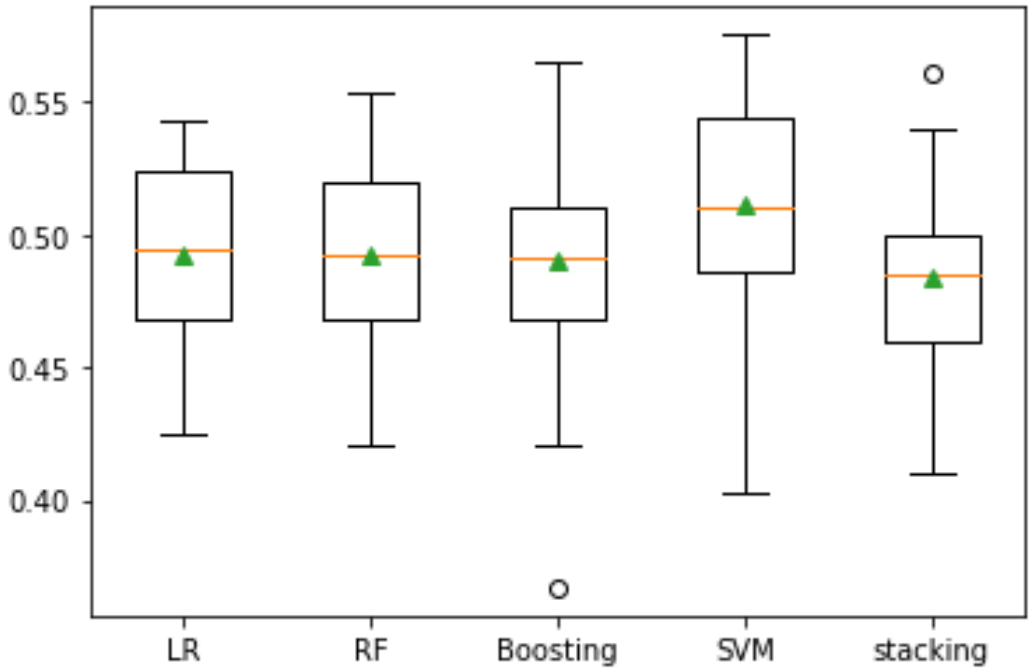


Figure 4.4: Classification result of stacking model on 30-minute light conditions for nine distinct genotype classes.

We found that LSTM model performed poorer than CNN models. For LSTM, the informational content of the vector will eventually be constrained by its fixed size. The information from the previous inputs runs the danger of being lost or diluted, especially for longer sequences which cause drop in model's performance. On the other hand, CNN models such as Conv2D, can automatically identify patterns and extract features from the input data, applying a series of convolution layers in successions such as weight-sharing filters and dimension-reducing pooling layers. These features are then passed to a series of dense layers for classification (or regression) which finally enable CNN model to do classification task more accurately and were able to outperform LSTM models in the classification of genotypes. Two-dimensional CNN (Conv2D) are commonly used in computer vision applications to interpret time as a spatial dimension.

For the ConvLSTM2D model, we incorporated the features of both the CNN and LSTM models together. Here, the most descriptive features in the data

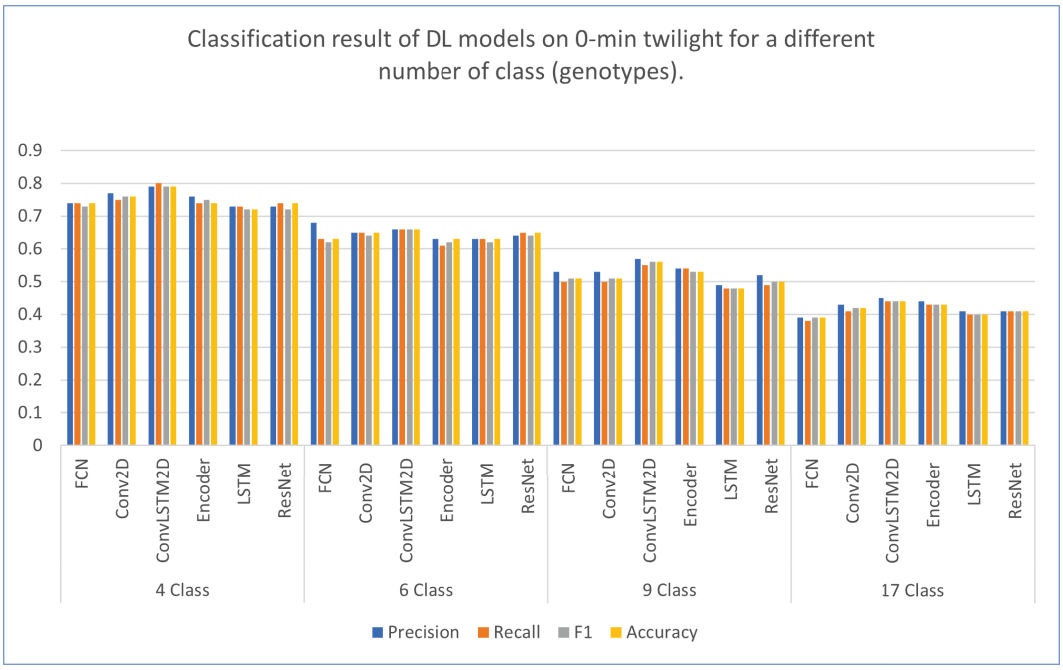


Figure 4.5: Classification result of DL models on 0-minute light conditions for a different number of classes (genotypes).

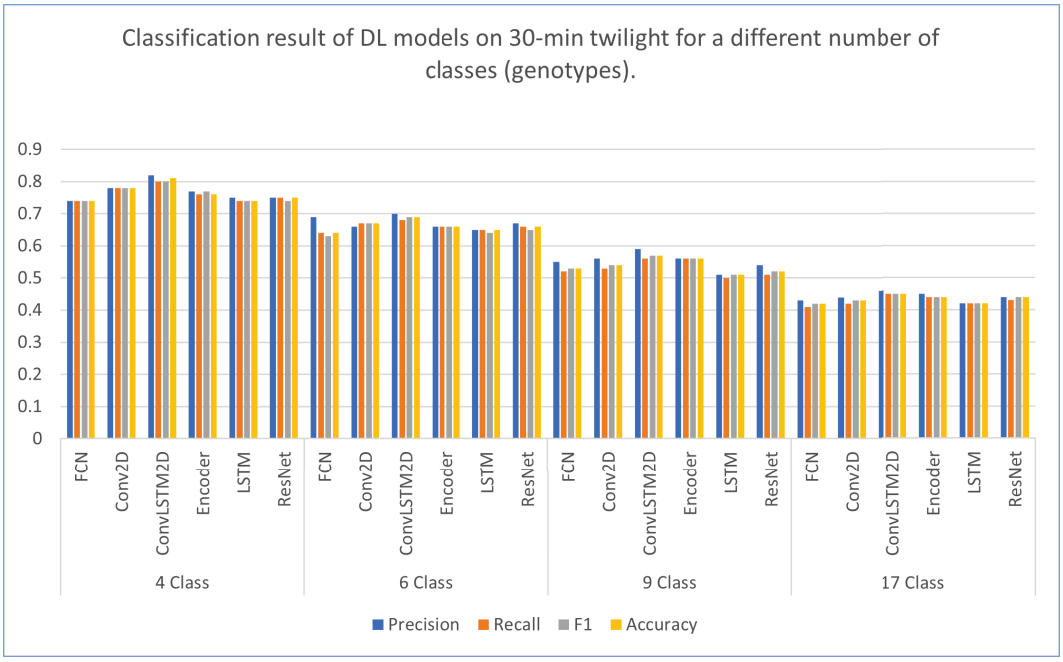


Figure 4.6: Classification result of DL models on 30-minute light conditions for a different number of classes (genotypes).

are automatically found and extracted by CNNs, while LSTM manages the dynamic behaviors occurring throughout plant growth and development. LSTM attempts to classify the plants by analyzing the sequence of features that are extracted from the daily time-series of plant growth and by considering their temporal variations, which in combination, provide better classification results than other models. The findings indicate that ConvLSTM2D is ideal for time-series across multiple scales, genotypes and growth conditions.

Table 4.4: Classification result of Deep learning (DL) models on 0-min twilight for a different number of classes (genotypes).

No of Class	Models	P	R	F1	Acc
4 Class	FCN	0.74	0.74	0.73	0.74
	Conv2D	0.77	0.75	0.76	0.76
	ConvLSTM2D	**0.79**	**0.80**	**0.79**	**0.79**
	Encoder	0.76	0.74	0.75	0.74
	LSTM	0.73	0.73	0.72	0.72
	ResNet	0.73	0.74	0.72	0.74
6 Class	FCN	0.68	0.63	0.62	0.63
	Conv2D	0.65	0.65	0.64	0.65
	ConvLSTM2D	**0.66**	**0.66**	**0.66**	**0.66**
	Encoder	0.63	0.61	0.62	0.63
	LSTM	0.63	0.63	0.62	0.63
	ResNet	0.64	0.65	0.64	0.65
9 Class	FCN	0.53	0.50	0.51	0.51
	Conv2D	0.53	0.50	0.51	0.51
	ConvLSTM2D	**0.57**	**0.55**	**0.56**	**0.56**
	Encoder	0.54	0.54	0.53	0.53
	LSTM	0.49	0.48	0.48	0.48
	ResNet	0.52	0.49	0.50	0.50
17 Class	FCN	0.39	0.38	0.39	0.39
	Conv2D	0.43	0.41	0.42	0.42
	ConvLSTM2D	**0.45**	**0.44**	**0.44**	**0.44**
	Encoder	0.44	0.43	0.43	0.43
	LSTM	0.41	0.40	0.40	0.40
	ResNet	0.41	0.41	0.41	0.41

To assess the performance of the models at an intra-group level, we examined whether or not the ConvLSTM2D model is able to classify different genotypes of "Phytochromes" group (*phyA* – *phyE*; class size = 5), "Pho-

Table 4.5: Classification result of Deep Learning (DL) models on 30-min twilight for a different number of classes (genotypes).

No of Class	Models	P	R	F1	Acc
4 Class	FCN	0.74	0.74	0.74	0.74
	Conv2D	0.78	0.78	0.78	0.78
	ConvLSTM2D	**0.82**	**0.80**	**0.80**	**0.81**
	Encoder	0.77	0.76	0.77	0.76
	LSTM	0.75	0.74	0.74	0.74
	ResNet	0.75	0.75	0.74	0.75
6 Class	FCN	0.69	0.64	0.63	0.64
	Conv2D	0.66	0.67	0.67	0.67
	ConvLSTM2D	**0.70**	**0.68**	**0.69**	**0.69**
	Encoder	0.66	0.66	0.66	0.66
	LSTM	0.65	0.65	0.64	0.65
	ResNet	0.67	0.66	0.65	0.66
9 Class	FCN	0.55	0.52	0.53	0.53
	Conv2D	0.56	0.53	0.54	0.54
	ConvLSTM2D	**0.59**	**0.56**	**0.57**	**0.57**
	Encoder	0.56	0.56	0.56	0.56
	LSTM	0.51	0.50	0.51	0.51
	ResNet	0.54	0.51	0.52	0.52
17 Class	FCN	0.43	0.41	0.42	0.42
	Conv2D	0.44	0.42	0.43	0.43
	ConvLSTM2D	**0.46**	**0.45**	**0.45**	**0.45**
	Encoder	0.45	0.44	0.44	0.44
	LSTM	0.42	0.42	0.42	0.42
	ResNet	0.44	0.43	0.44	0.44

totropins" group (*phot1, phot2, phot1/2*; class size = 3) and "Cryptochromes" group (*cry1, cry2 cry1/2*; class size = 3). We evaluated the performance by presenting the confusion matrix (Figure 4.7). We selected 3 classes from 3 different light conditions to make a total of nine potential classes for each group. From this analysis, we extracted precision, recall, F1-score, and accuracy of 67%, 66%, 66%, and 66%, respectively for "Phototropin" group, while achieving precision, recall, F1-score, and accuracy of 57%, 56%, 56%, and 56% respectively for the "Cryptochrome" group (Table 4.7). These suggest that ConvLSTM2D successfully classified different genotypes under different twilight conditions using multi-scale time-series data, and based on the confusion matrix, we can readily identify which genotypes are classified/misclassified

Table 4.6: Classification result of Deep learning (DL) models on 90-min twilight for a different number of classes (genotypes).

No of Class	Models	P	R	F1	Acc
4 Class	FCN	0.74	0.74	0.73	0.74
	Conv2D	0.77	0.76	0.77	0.77
	ConvLSTM2D	**0.80**	**0.79**	**0.80**	**0.80**
	Encoder	0.76	0.76	0.75	0.75
	LSTM	0.74	0.73	0.74	0.73
	ResNet	0.73	0.73	0.72	0.73
6 Class	FCN	0.66	0.64	0.64	0.64
	Conv2D	0.65	0.65	0.64	0.65
	ConvLSTM2D	**0.68**	**0.67**	**0.68**	**0.67**
	Encoder	0.64	0.63	0.64	0.63
	LSTM	0.62	0.61	0.62	0.62
	ResNet	0.65	0.64	0.64	0.64
9 Class	FCN	0.53	0.50	0.51	0.50
	Conv2D	0.53	0.52	0.52	0.52
	ConvLSTM2D	**0.57**	**0.55**	**0.56**	**0.56**
	Encoder	0.54	0.54	0.53	0.53
	LSTM	0.49	0.49	0.49	0.49
	ResNet	0.52	0.50	0.50	0.51
17 Class	FCN	0.40	0.39	0.40	0.40
	Conv2D	0.43	0.41	0.42	0.42
	ConvLSTM2D	**0.45**	**0.44**	**0.44**	**0.44**
	Encoder	0.44	0.43	0.43	0.43
	LSTM	0.42	0.42	0.41	0.41
	ResNet	0.41	0.40	0.41	0.41

to which class. The performance analysis by ConvLSTM2D model for each group of related photoreceptor deficient plants is shown in Tables 4.4, 4.5, and 4.6 from which we observed several similarities between the genotype classes inside each group. For instance, we see a more similar intra-/inter-day growth pattern between plants of genotypes *phot2 and phot1/2*, relative to *phot1* inside the "Phototropins" group, with *phot2* plants consistently miscategorized as *phot1/2*.

.

Phototropins are particularly interesting as they control leaf movement and positioning in response to blue light [34]. Compared to *wt* plants, *phot1 and phot2* plants possess phototropic bending towards high and low fluence

Figure 4.7: Performance of ConvLSTM2D model on different groups of plants for different twilights (0, 30, and 90-minutes).

blue light, respectively, while *phot1/2* plants have no phototropic responses to either low or high blue light. By collecting time-series data over multiple timescales, our surface area data reveals specific intra-day grow patterns for each genotype (Figure 4.8). In most genotypes, area data is not linear over the course of a day, but rather forms a peak at mid-day. This suggests that changes in surface area may result from leaf movement across the day. Correspondingly, we hypothesize that this creates a signature feature that the ConvLSTM2D model can be successfully used to distinguish unique genotypes. In the case of the phototropins, we can see that *phot2* plant area is similar to that of *phot1/2*, with both being relatively "flat" across the day in a twilight-dependent manner. This is especially apparent in the 90min twilight condition, where *phot2* and *phot1/2* possess minimal variation, which correlates with their mis-categorization, suggesting that changes in leaf area, mediated by leaf movement, create a unique feature over time that DL models can utilize to detect differences in plant genotypes. In particular, the ConvLSTM2D model using time-series data across multiple scales represents the best model for differentiating genotypes grown under different light conditions. From the confusion matrix, we can clearly see that *phot2* type of plants are misclassified into *phot1/2* type of plants. For example, phot2_0_min has a misclassification of 0.12 into phot1/2_0_min. Here, the model predicted 12% of *phot2* plants as *phot1/2* plants mistakenly (False Negative). Again, phot2_30_min

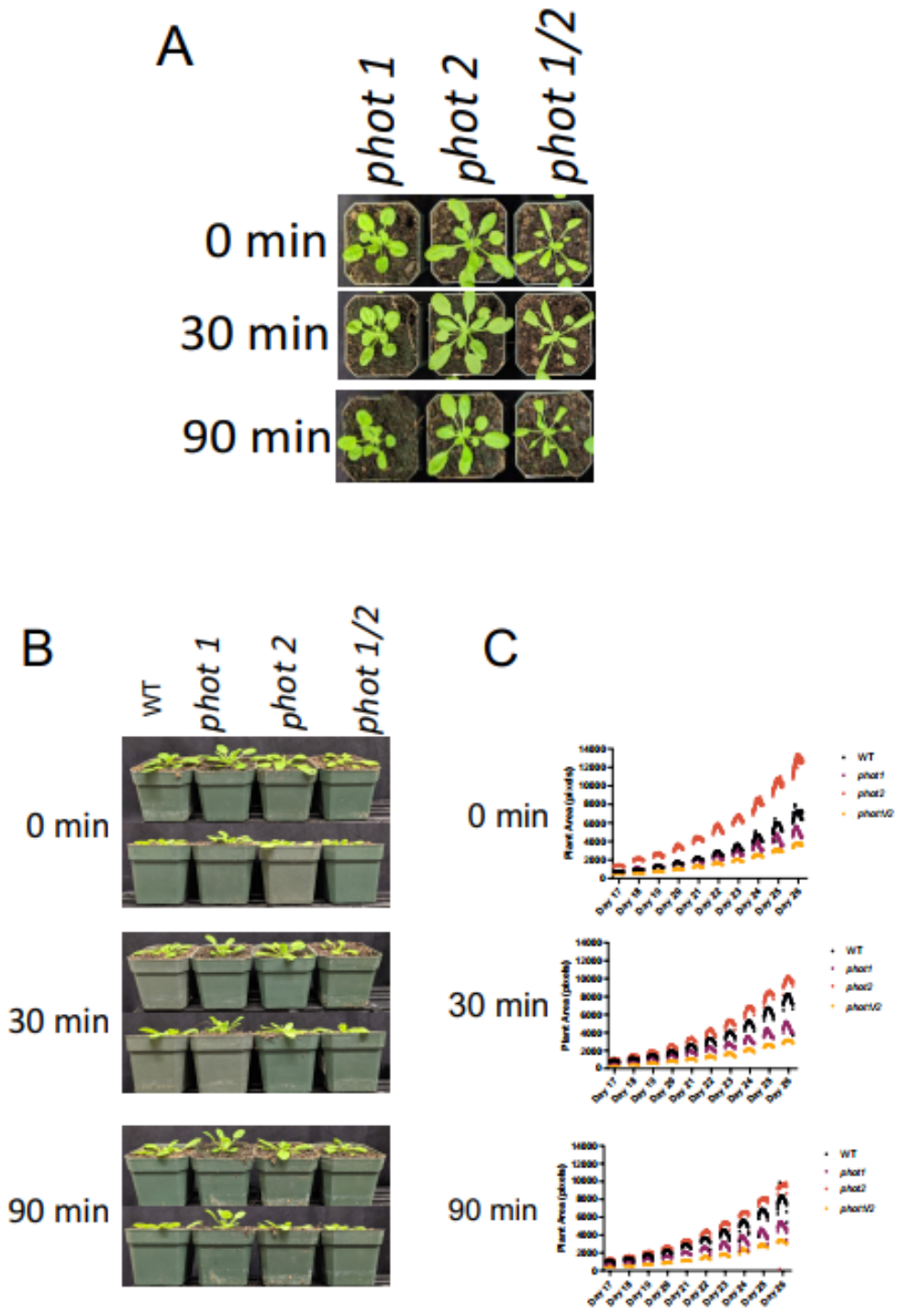


Figure 4.8: Phototropins mutants details. (A) Pictures from the top of phototropins mutants *phot1, phot2, and phot1/2* at different light treatment. (B) Pictures from the side of *phot1, phot2 and phot1/2* at different light treatment.(C) Plant area in pixels of phototropins mutants calculated with PlantCV.

plants get misclassified as phot1/2_30_min plants with a misclassification rate of 0.31. Here again, 31% plant of *phot2* plants are getting misclassified into phot1/2. Again, phot1/2_0_min plants have a false negative rate of 0.24 with phot2_0_min plants which means 24% of *phot1/2* plants are falsely predicted as *phot1/2* plants.

Table 4.7: Performance of ConvLSTM2D model on different groups of plants for different twilights (0, 30, and 90-minutes).

Group	Time	P	R	F1	Accuracy
Phytochrome	0 min	0.61	0.61	0.61	0.61
	30 min	0.66	0.66	0.66	0.66
	90 min	0.61	0.61	0.61	0.61
Phototropins	0 min	0.72	0.72	0.72	0.72
	30 min	0.74	0.73	0.73	0.73
	90 min	0.71	0.70	0.71	0.71
ztl/fkf1/lkp2	0 min	0.66	0.66	0.66	0.66
	30 min	0.66	0.66	0.66	0.66
	90 min	0.64	0.65	0.64	0.65
Cryptochrome	0 min	0.66	0.66	0.66	0.66
	30 min	0.67	0.66	0.67	0.67
	90 min	0.64	0.64	0.64	0.64

4.0.1 Statistical Testing

We conducted hypothesis testing using the 5x2 corss-validation procedure With MLxtend to assess the performance of the models. One approach is to evaluate each model on the same k-fold cross-validation split of the data (e.g. using the same random number seed to split the data in each case) and calculate a score for each split. This would give a sample of 10 scores for 10-fold cross-validation. As a result of each algorithm employing the identical treatment (rows of data), each score can be compared using a paired statistical hypothesis test. One option is to employ the paired student t-test. The fact that each evaluation of the model is not independent presents a challenge when applying the Paired Student's t-Test in this situation. This is due to the fact that the same rows of data are used repeatedly to train the data; in fact, each time, save for the instance where a row of data is utilized in the hold-out test fold. Because of this lack of impartiality in the judgement, the Paired Student's t-Test is biased in favour of optimism. The statistical test can be modified to account for the lack of independence. Additionally, the procedure's folds and repeats can be adjusted to generate a representative sample of model performance that applies well to a variety of issues and methods. Specifically, the so-called 5x2-fold cross-validation, which entails two-fold cross-validation with five repetitions. After applying the procedure between SVM and LR (Figure 4.9), and SVM and RF (Figure 4.10), we proved that in all scenarios SVM outperformed all other methods in term of performance metrics. However, as the data distribution are same, there is no significant difference among the performance of the models.

Further, we experimented with One-Way ANOVA (also known as "analysis of variance") to find out whether there exists a statistically significant difference between the mean values of the model's performance.

Hypothesis involved:

H0 (null hypothesis): a1 = a2 = a3 = ... = ak (It implies that the means of all the population are equal)

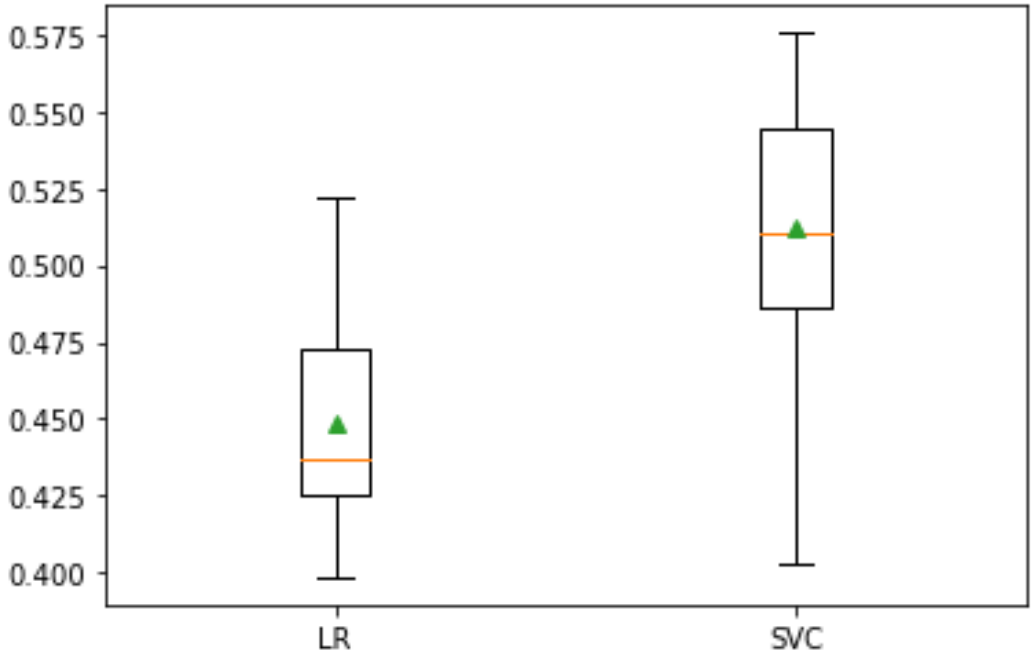


Figure 4.9: 5 x 2 Cross-validation procedure to prove models' performance (LR and SVC) using 30-min twilight dataset.

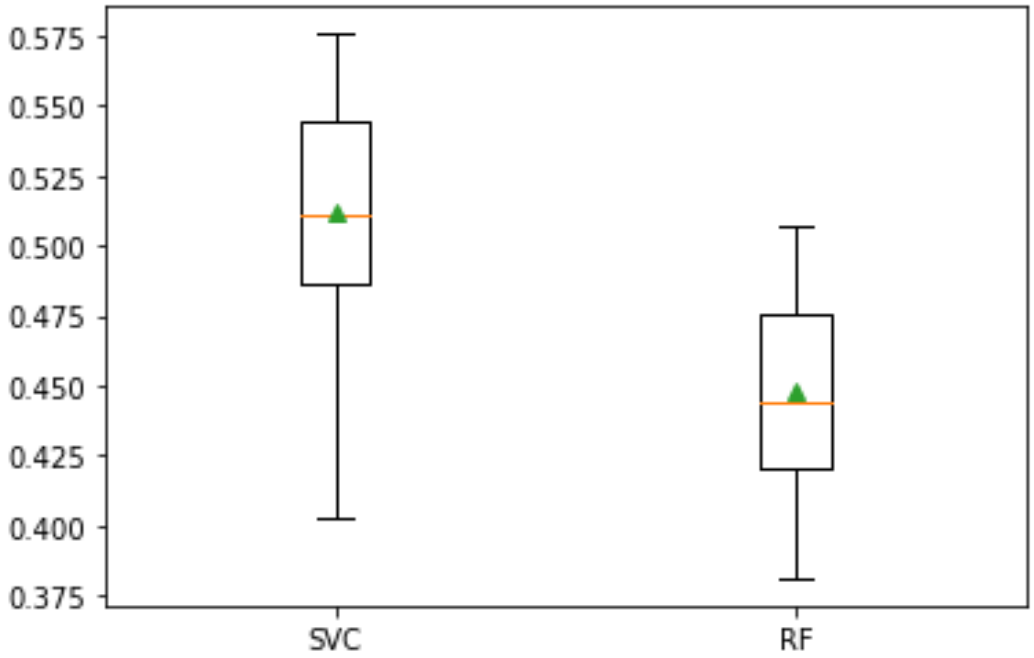


Figure 4.10: 5 x 2 Cross-validation procedure to prove models' performance (SVC and RF) using 30-min twilight dataset.

H1 (alternative hypothesis): It states that there will be at least one population mean that differs from the rest.

As we have more than 2 models for both cases traditional and deep learning models, we used one-way anova for the analysis. We have our average precision, recall, f1-score, and accuracy values of all the traditional and deep learning models after cross-validation. We tried to determine if there exists a difference in the performance among the models or not. We took into consideration all the models' average performance (precision, recall, f1-score, and accuracy) on 30-min twilight for 6 genotype classes. After considering all the traditional ML and deep learning models in our analyses, we found the p-values of 5.3179e-06 and 0.0017 respectively. Since the p-value is less than 0.05 for both cases, we would reject the null hypothesis. This suggests that we have enough evidence to assert that there is a distinction between the average performance of conventional machine learning models and deep learning models.

In addition, as we claimed SVM to be the best performer among the traditional ML models, there should exist significant difference among the the mean performance of SVM with other models. We then compared SVM with each model and got the p-values of 0.0001, 0.0005, 0.010 respectively for Boosting, LR, and RF models. Since the p-values are less than 0.05 in every case, we can reject the null hypothesis and imply that the performance distribution of SVM is different than other models and their exist significant differences. So, we can say that SVM model is better than all other traditional models but only with p-values 0.05. Again, ConvLSTM2D performed the best among the deep learning models. We compared the mean performance of ConvLSTM2D model separately with all the model and got the p-values of 0.03, 0.002, 0.003, 0.0001, 0.003 for FCN, ResNet, Encoder, LSTM, Conv2D models respectively. Here, p-values are less than 0.05 in every case, we can reject the null hypothesis and imply that the performance distribution of ConvLSTM2D is different than other models and there exist significant differences. So, we can say that ConvLSTM2D model is better than all other traditional models but only with

p-values 0.05.

4.1 "wt" plants genotype class similarity

We conducted experiments with plants of the *"wt"* genotype in an effort to determine the percentage of correctly predicted members of a certain class and the percentage of incorrectly classified members of other genotype classes. We conducted two experiments using two deep learning models - ConvLSTM2D and Encoder.

In the experiment 1 (Figure 4.12), we sampled out a total of 1250 plants including all the 17 genotype classes. We tried to find out how many other genotype plants got incorrectly labeled (False Positive) as *"wt"* genotype. We made an effort by using ConvLSTM2D model to determine how many *"wt"* plants were incorrectly labeled as other plants. Among 1250 plants, 130 plants truly belonged to *"wt"* genotype class. Encoder predicted 121 plants as *"wt"* and among them it truly predicted 63 plants as *"wt"* out of 130 plants and remaining 58 plants were incorrectly labeled as *"wt"* genotype. We got a recall score of 63/130 = 0.484 or 48.4% and precision score 63/(63+58) = 0.52 or 52%. Here, 67 *"wt"* plants are misclassified as other genotype. From further analysis, we found the highest misclassification rate with 13.5% in *"cry2"* type plants. Among the 67 misclasssified plants, Among the misclassification, 16.7% of "cry2" plants are misclassified as *"wt"* type, which is the highest misclassification genotype. Plant genotypes *"phot2"* and *"ztl"* came in second and third, respectively, to *"cry2"*, predicted falsely as "wt" genotype with misclassification rates of 11.7% and 10.9%.

In experiment 2 (Figure 4.12), we did the opposite prediction of experiment 1, using Encoder deep learning model. We made an effort to determine how many *"wt"* plants were incorrectly labeled as other plants. Among 1250 plants, 130 plants truly belonged to *"wt"* genotype class. Encoder predicted 63 plants as *"wt"* genotype class and remaining 67 "wt" plants were incorrectly labeled as other genotype plants. Again, just like experiment 1, we found the highest misclassification rate with 13.5% in "cry2" type plants. Among the 67 misclasssified plants, 9 plants of *"wt"* type were incorrectly predicted as

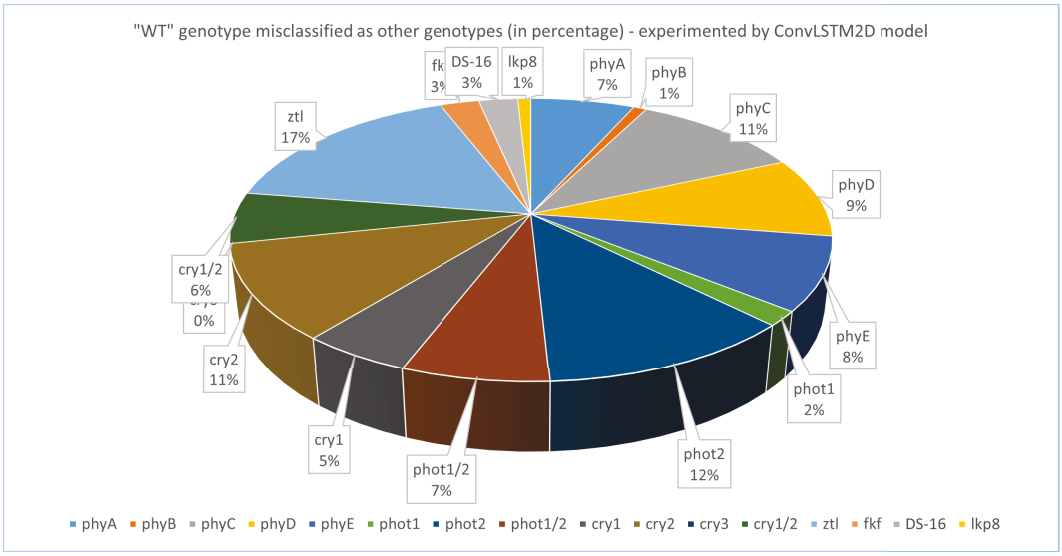


Figure 4.11: Other genotypes misclassified as *"wt"* genotype- experiment by ConvLSTM2D model.

"cry2" type, 8 *"wt"* plants were misclassified as *"phot2"* type, and another 8 *"wt"* plants were misclassified as *"ztl"* type with the misclassification rate of 12%.

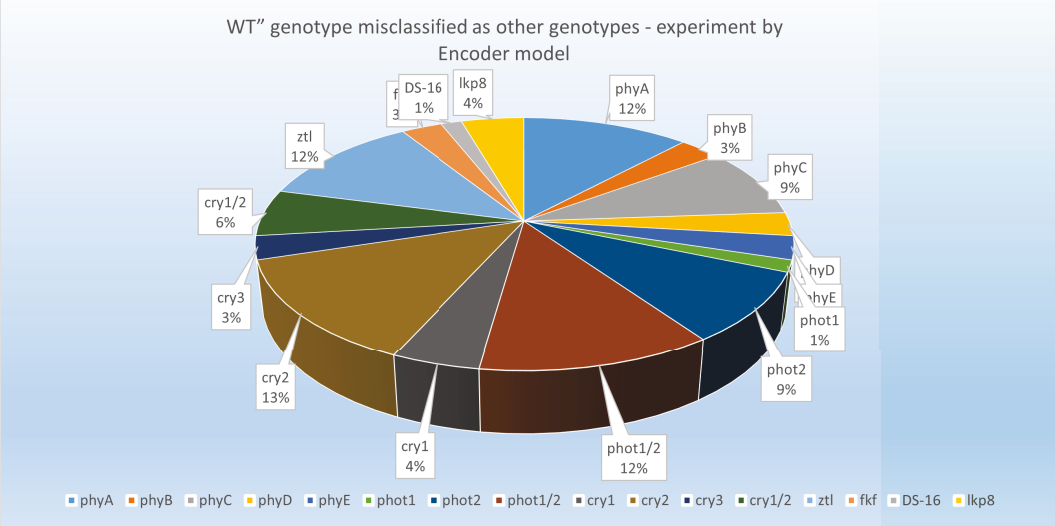


Figure 4.12: *"wt"* genotype misclassified as other genotypes - experiment by Encoder model

Chapter 5

Conclusion

In this study, we evaluated multiple traditional ML and most recent DL algorithms for their ability to perform genotype classification and prediction for Arabidopsis thaliana plants using a combined intra- and inter-day time-series growth dataset derived from 17 different genotypes subjected to three different twilight growth conditions. Each of the tested ML and DL algorithms exhibits a capacity for capturing subtle growth features which would otherwise escape manual inspection. We found that DL algorithms perform comparatively better, likely owing to their interpretation of time-series data which is typically mapped into independent dimensions by classical ML algorithms such as SVM. Among the DL algorithms we tested, our results find that ConvLSTM2D outperformed all the other DL algorithms in all measurements including precision, recall, F1-score, and accuracy, using the leaf area values extracted from the plant growth images for genotype classification/prediction. On the other hand, SVM also performed exceptionally well in handling time series data. Critically, our findings, which successfully integrated time-series growth data across scales (intra- and inter-day), genotypes, and growth conditions represent a significant advancement, setting a new foundational baseline from which more variables can be integrated in order to assess more complex plant traits. However, there are several areas for future work. First, our study focused on a specific plant genotype, and it would be valuable to expand our analysis to include a broader range of plant genotypes. Second, our study utilized only the area data, and we expect that more plant growth characteristics

such as perimeter, leaf greenness, leaves count will further improve genotype classification by machine learning models. We aim at developing easy-to-use computer vision programs for extracting these additional features to improve genotype-phenotype connections by deep learning algorithms in the plant sciences.

References

[1] M. Ahmad, N. Grancher, M. Heil, *et al.*, "Action spectrum for cryptochrome-dependent hypocotyl growth inhibition in arabidopsis," *Plant Physiology*, vol. 129, no. 2, pp. 774–785, 2002.

[2] A. Akhtar, A. Khanum, S. A. Khan, and A. Shaukat, "Automated plant disease analysis (apda): Performance comparison of machine learning techniques," in *2013 11th International Conference on Frontiers of Information Technology*, IEEE, 2013, pp. 60–65.

[3] W. Albattah, M. Nawaz, A. Javed, M. Masood, and S. Albahli, "A novel deep learning method for detection and classification of plant diseases," *Complex & Intelligent Systems*, vol. 8, no. 1, pp. 507–524, 2022.

[4] S. Anubha Pearline, V. Sathiesh Kumar, and S. Harini, "A study on plant recognition using conventional image processing and deep learning approaches," *Journal of Intelligent & Fuzzy Systems*, vol. 36, no. 3, pp. 1997–2004, 2019.

[5] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances," *Data mining and knowledge discovery*, vol. 31, no. 3, pp. 606–660, 2017.

[6] M. H. Bhavsar and A. Ganatra, "Radial basis polynomial kernel (rbpk): A generalized kernel for support vector machine," *International Journal of Computer Science and Information Security (IJCSIS)*, vol. 14, no. 4, 2016.

[7] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[8] J. Brownlee, *Use early stopping to halt the training of neural networks at the right time*, Aug. 2020. [Online]. Available: `https://machinelearningmastery.com/how-to-stop-training-deep-neural-networks-at-the-right-time-using-early-stopping/`.

[9] M. Campos-Taberner, F. J. Garcıa-Haro, B. Martınez, *et al.*, "Understanding deep learning in land use classification based on sentinel-2 time series," *Scientific reports*, vol. 10, no. 1, pp. 1–12, 2020.

[10] J. C.-W. Chan and D. Paelinckx, "Evaluation of random forest and adaboost tree-based ensemble classification and spectral band selection for ecotope mapping using airborne hyperspectral imagery," *Remote Sensing of Environment*, vol. 112, no. 6, pp. 2999–3011, 2008.

[11] D. Chen, K. Neumann, S. Friedel, *et al.*, "Dissecting the phenotypic components of crop plant growth and drought responses based on high-throughput image analysis," *The plant cell*, vol. 26, no. 12, pp. 4636–4655, 2014.

[12] H. Choi, D. Yeo, S. Kwon, and Y. Kim, "Gene selection and prediction for cancer classification using support vector machines with a reject option," *Computational statistics & data analysis*, vol. 55, no. 5, pp. 1897–1908, 2011.

[13] G. Ciaburro, V. K. Ayyadevara, and A. Perrier, *Hands-on machine learning on google cloud platform*. [Online]. Available: `https://www.oreilly.com/library/view/hands-on-machine-learning/9781788393485/fd5b8a44-e9d3-4c19-bebb-c2fa5a5ebfee.xhtml`.

[14] N. Creux and S. Harmer, "Circadian rhythms in plants," *Cold Spring Harbor Perspectives in Biology*, vol. 11, no. 9, a034611, 2019.

[15] H. Crisóstomo de Castro Filho, O. Abılio de Carvalho Júnior, O. L. Ferreira de Carvalho, *et al.*, "Rice crop detection using lstm, bi-lstm, and machine learning models from sentinel-1 time series," *Remote Sensing*, vol. 12, no. 16, p. 2655, 2020.

[16] M. Dileep and P. Pournami, "Ayurleaf: A deep learning approach for classification of medicinal plants," in *TENCON 2019-2019 IEEE Region 10 Conference (TENCON)*, IEEE, 2019, pp. 321–325.

[17] S. El-Din El-Assal, C. Alonso-Blanco, A. J. Peeters, C. Wagemaker, J. L. Weller, and M. Koornneef, "The role of cryptochrome 2 in flowering in arabidopsis," *Plant Physiology*, vol. 133, no. 4, pp. 1504–1516, 2003.

[18] R. Dıaz-Uriarte and S. Alvarez de Andrés, "Gene selection and classification of microarray data using random forest," *BMC bioinformatics*, vol. 7, no. 1, pp. 1–13, 2006.

[19] S. Eberhard, G. Finazzi, and F.-A. Wollman, "The dynamics of photosynthesis," *Annual review of genetics*, vol. 42, pp. 463–515, 2008.

[20] K. Eckle and J. Schmidt-Hieber, "A comparison of deep networks with relu activation function and linear spline-type methods," *Neural Networks*, vol. 110, pp. 232–242, 2019.

[21] M. Endo, Y. Tanigawa, T. Murakami, T. Araki, and A. Nagatani, "Phytochrome-dependent late-flowering accelerates flowering through physical interactions with phytochrome b and constans," *Proceedings of the National Academy of Sciences*, vol. 110, no. 44, pp. 18 017–18 022, 2013.

[22] K. A. Franklin, V. S. Larner, and G. C. Whitelam, "The signal transducing photoreceptors of plants," *International Journal of Developmental Biology*, vol. 49, no. 5-6, pp. 653–664, 2004.

[23] D. P. Fraser, P. E. Panter, A. Sharma, B. Sharma, A. N. Dodd, and K. A. Franklin, "Phytochrome a elevates plant circadian-clock components to suppress shade avoidance in deep-canopy shade," *Proceedings of the National Academy of Sciences*, vol. 118, no. 27, e2108176118, 2021.

[24] V. C. Galvão and C. Fankhauser, "Sensing the light environment in plants: Photoreceptors and early signaling steps," *Current opinion in neurobiology*, vol. 34, pp. 46–53, 2015.

[25] B. Gaonkar and C. Davatzikos, "Analytic estimation of statistical significance maps for support vector machine based multi-variate image analysis and classification," *Neuroimage*, vol. 78, pp. 270–283, 2013.

[26] M. A. Gehan, N. Fahlgren, A. Abbasi, *et al.*, "Plantcv v2: Image analysis software for high-throughput plant phenotyping," *PeerJ*, vol. 5, e4088, 2017.

[27] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*, Ieee, 2013, pp. 6645–6649.

[28] H. Guo, H. Yang, T. C. Mockler, and C. Lin, "Regulation of flowering time by arabidopsis photoreceptors," *Science*, vol. 279, no. 5355, pp. 1360–1363, 1998.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[32] Z. Ibrahim, N. Sabri, and N. N. A. Mangshor, "Leaf recognition using texture features for herbal plant identification," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 9, no. 1, pp. 152–156, 2018.

[33] N. Inagaki, K. Kinoshita, T. Kagawa, *et al.*, "Phytochrome b mediates the regulation of chlorophyll biosynthesis through transcriptional regulation of chlh and gun4 in rice seedlings," *PLoS One*, vol. 10, no. 8, e0135408, 2015.

[34] S.-i. Inoue, T. Kinoshita, and K.-i. Shimazaki, "Possible involvement of phototropins in leaf movement of kidney bean in response to blue light," *Plant physiology*, vol. 138, no. 4, pp. 1994–2004, 2005.

[35] Y. Jiang and C. Li, "Convolutional neural networks for image-based high-throughput plant phenotyping: A review," *Plant Phenomics*, vol. 2020, 2020.

[36] S. Kaur and P. Kaur, "Plant species identification based on plant leaf using computer vision and machine learning techniques," *Journal of Multimedia Information System*, vol. 6, no. 2, pp. 49–60, 2019.

[37] Y. Kimura, I. Kimura, and T. Kanegae, "Phototropins of the moss physcomitrella patens function as blue-light receptors for phototropism in arabidopsis," *Plant signaling & behavior*, vol. 13, no. 10, e1525995, 2018.

[38] S. Kolhar and J. Jagtap, "Spatio-temporal deep neural networks for accession classification of arabidopsis plants using image sequences," *Ecological Informatics*, vol. 64, p. 101 334, 2021.

[39] Y. Kong and Y. Zheng, "Phytochrome contributes to blue-light-mediated stem elongation and associated shade-avoidance response in mature arabidopsis plants," *bioRxiv*, 2020.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[41] G.-H. Kwak, M.-G. Park, C.-W. Park, *et al.*, "Combining 2d cnn and bidirectional lstm to consider spatio-temporal features in crop classification," *Korean Journal of Remote Sensing*, vol. 35, no. 5_1, pp. 681–692, 2019.

[42] P. Lariguet and C. Dunand, "Plant photoreceptors: Phylogenetic overview," *Journal of Molecular Evolution*, vol. 61, no. 4, pp. 559–569, 2005.

[43] C. Lee, M. Li, A. Feke, W. Liu, A. Saffer, and J. Gendron, *Gigantea recruits the ubp12 and ubp13 deubiquitylases to regulate accumulation of the ztl photoreceptor complex. nat. commun. 10: 3750*, 2019.

[44] S. H. Lee, C. S. Chan, S. J. Mayo, and P. Remagnino, "How deep learning extracts and learns leaf features for plant classification," *Pattern Recognition*, vol. 71, pp. 1–13, 2017.

[45] S. H. Lee, Y. L. Chang, C. S. Chan, and P. Remagnino, "Plant identification system based on a convolutional neural network for the lifeclef 2016 plant classification task.," *CLEF (Working Notes)*, vol. 1, pp. 502–510, 2016.

[46] F.-W. Li and S. Mathews, "Evolutionary aspects of plant photoreceptors," *Journal of plant research*, vol. 129, no. 2, pp. 115–122, 2016.

[47] T. Li, M. Hua, and X. Wu, "A hybrid cnn-lstm model for forecasting particulate matter (pm2. 5)," *Ieee Access*, vol. 8, pp. 26 933–26 940, 2020.

[48] M. Litvak, S. Divekar, and I. Rabaev, "Urban plants classification using deep-learning methodology: A case study on a new dataset," *Signals*, vol. 3, no. 3, pp. 524–534, 2022.

[49] J. Liu, J. Shang, B. Qian, *et al.*, "Crop yield estimation using time-series modis data and the effects of cropland masks in ontario, canada," *Remote Sensing*, vol. 11, no. 20, p. 2419, 2019.

[50] L. Lopez, C. Fasano, G. Perrella, and P. Facella, "Cryptochromes and the circadian clock: The story of a very complex relationship in a spinning world," *Genes*, vol. 12, no. 5, p. 672, 2021.

[51] P. Lymperopoulos, J. Msanne, and R. Rabara, "Phytochrome and phytohormones: Working in tandem for plant growth and development," *Frontiers in Plant Science*, vol. 9, p. 1037, 2018.

[52] Z. Mohammed Amean, T. Low, C. McCarthy, and N. Hancock, "Automatic plant branch segmentation and classification using vesselness measure," in *Proceedings of the Australasian Conference on Robotics and Automation (ACRA 2013)*, Australasian Robotics and Automation Association, 2013, pp. 1–9.

[53] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in plant science*, vol. 7, p. 1419, 2016.

[54] S. Nagano, K. Guan, S. M. Shenkutie, *et al.*, "Structural insights into photoactivation and signalling in plant phytochromes," *Nature Plants*, vol. 6, no. 5, pp. 581–588, 2020.

[55] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv preprint arXiv:1811.03378*, 2018.

[56] Z. Olaofe, "Assessment of lstm, conv2d and convlstm2d prediction models for long-term wind speed and direction regression analysis," 2021.

[57] T. Pahikkala, K. Kari, H. Mattila, *et al.*, "Classification of plant species from images of overlapping leaves," *Computers and Electronics in Agriculture*, vol. 118, pp. 186–192, 2015.

[58] X.-Y. Pan and H.-B. Shen, "Robust prediction of b-factor profile from sequence using two-stage svr based on random forest feature selection," *Protein and peptide letters*, vol. 16, no. 12, pp. 1447–1454, 2009.

[59] D. R. Pereira, J. P. Papa, G. F. R. Saraiva, and G. M. Souza, "Automatic classification of plant electrophysiological responses to environmental stimuli using machine learning and interval arithmetic," *Computers and Electronics in Agriculture*, vol. 145, pp. 35–42, 2018.

[60] R. Podolec, E. Demarsy, and R. Ulm, "Perception and signaling of ultraviolet-b radiation in plants," *Annual Review of Plant Biology*, vol. 72, pp. 793–822, 2021.

[61] A. M. Prasad, L. R. Iverson, and A. Liaw, "Newer classification and regression tree techniques: Bagging and random forests for ecological prediction," *Ecosystems*, vol. 9, no. 2, pp. 181–199, 2006.

[62] O. J. Prieto, C. J. Alonso-González, and J. J. Rodrıguez, "Stacking for multivariate time series classification," *Pattern Analysis and Applications*, vol. 18, no. 2, pp. 297–312, 2015.

[63] M. M. Rahaman, M. A. Ahsan, and M. Chen, "Data-mining techniques for image-based plant phenotypic traits identification and classification," *Scientific reports*, vol. 9, no. 1, p. 19 526, 2019.

[64] J. Rivers, N. Warthmann, B. J. Pogson, and J. O. Borevitz, "Genomic breeding for food, environment and livelihoods," *Food Security*, vol. 7, no. 2, pp. 375–382, 2015.

[65] N. C. Rockwell, Y.-S. Su, and J. C. Lagarias, "Phytochome structure and signaling mechanisms," *Annual review of plant biology*, vol. 57, p. 837, 2006.

[66] A. Romanowski, J. J. Furniss, E. Hussain, and K. J. Halliday, "Phytochrome regulates cellular response plasticity and the basic molecular machinery of leaf development," *Plant physiology*, vol. 186, no. 2, pp. 1220–1239, 2021.

[67] M. H. Saleem, J. Potgieter, and K. M. Arif, "Plant disease detection and classification by deep learning," *Plants*, vol. 8, no. 11, p. 468, 2019.

[68] R. Samsudin, A. Shabri, and P. Saad, "A comparison of time series forecasting using support vector machine and artificial neural network model," *Journal of applied sciences*, vol. 10, no. 11, pp. 950–958, 2010.

[69] A. Sancar, "Structure and function of dna photolyase and cryptochrome blue-light photoreceptors," *Chemical reviews*, vol. 103, no. 6, pp. 2203–2238, 2003.

[70] S. E. Sanchez, M. L. Rugnone, and S. A. Kay, "Light perception: A matter of time," *Molecular plant*, vol. 13, no. 3, pp. 363–385, 2020.

[71] M. Schikora, A. Schikora, K.-H. Kogel, W. Koch, and D. Cremers, "Probabilistic classification of disease symptoms caused by salmonella on arabidopsis plants," *INFORMATIK 2010. Service Science–Neue Perspektiven für die Informatik. Band 2*, 2010.

[72] J. Serrà, S. Pascual, and A. Karatzoglou, "Towards a universal neural network encoder for time series.," in *CCIA*, 2018, pp. 120–129.

[73] V. Sharma, R. Khemnar, R. Kumari, and B. R. Mohan, "Time series with sentiment analysis for stock price prediction," in *2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT)*, IEEE, 2019, pp. 178–181.

[74] T. Shinomura, A. Nagatani, J. Chory, and M. Furuya, "The induction of seed germination in arabidopsis thaliana is regulated principally by phytochrome b and secondarily by phytochrome a," *Plant physiology*, vol. 104, no. 2, pp. 363–371, 1994.

[75] V. Sineshchekov, O. Belyaeva, and A. Sudnitsin, "Up-regulation by phytochrome a of the active protochlorophyllide, pchlide655, biosynthesis in dicots under far-red light," *Journal of Photochemistry and Photobiology B: Biology*, vol. 74, no. 1, pp. 47–54, 2004.

[76] T. Subetha, R. Khilar, and M. S. Christo, "A comparative analysis on plant pathology classification using deep learning architecture–resnet and vgg19," *Materials Today: Proceedings*, 2021.

[77] S. Taghavi Namin, M. Esmaeilzadeh, M. Najafi, T. B. Brown, and J. O. Borevitz, "Deep phenotyping: Deep learning for temporal phenotype/genotype classification," *Plant methods*, vol. 14, no. 1, pp. 1–14, 2018.

[78] J. R. Ubbens and I. Stavness, "Corrigendum: Deep plant phenomics: A deep learning platform for complex plant phenotyping tasks," *Frontiers in plant science*, vol. 8, p. 2245, 2018.

[79] S. Varela, T. Pederson, C. J. Bernacchi, and A. D. Leakey, "Understanding growth dynamics and yield prediction of sorghum using high temporal resolution uav imagery time series and machine learning," *Remote Sensing*, vol. 13, no. 9, p. 1763, 2021.

[80] C. Wang, M. Sun, L. Liu, W. Zhu, P. Liu, and X. Li, "A high-accuracy genotype classification approach using time series imagery," *Biosystems Engineering*, vol. 220, pp. 172–180, 2022.

[81] X. Wang, X. Gao, Y. Liu, S. Fan, and Q. Ma, "Progress of research on the regulatory pathway of the plant shade-avoidance syndrome," *Frontiers in plant science*, vol. 11, p. 439, 2020.

[82] Z. Wang, H. Li, Y. Zhu, and T. Xu, "Review of plant identification based on image processing," *Archives of Computational Methods in Engineering*, vol. 24, no. 3, pp. 637–654, 2017.

[83] K. Yang, W. Zhong, and F. Li, "Leaf segmentation and classification with a complicated background using deep learning," *Agronomy*, vol. 10, no. 11, p. 1721, 2020.

[84] X. Yang and T. Guo, "Machine learning in plant disease research," *March*, vol. 31, p. 1, 2017.

[85] R. Yasrab, M. P. Pound, A. P. French, and T. P. Pridmore, "Phenomnet: Bridging phenotype-genotype gap: A cnn-lstm based automatic plant root anatomization system," *bioRxiv*, 2020.

[86] M. Zhang, H. Zhang, X. Li, Y. Liu, Y. Cai, and H. Lin, "Classification of paddy rice using a stacked generalization approach and the spectral mixture method based on modis time series," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 2264–2275, 2020.

[87] Z. Zhang, "Improved adam optimizer for deep neural networks," in *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, Ieee, 2018, pp. 1–2.

[88] Z. Zhang, *Boosting algorithms explained*, Aug. 2019. [Online]. Available: `https://towardsdatascience.com/boosting-algorithms-explained-d38f56ef3f30`.